POLITECNICO
MILANO 1863

POLITECNICO DI MILANO
DEPARTMENT OF AEROSPACE SCIENCE AND TECHNOLOGY
DOCTORAL PROGRAMME IN AEROSPACE ENGINEERING

Conservative interpolation-free mesh adaptation for three-dimensional aeroelastic simulations in unsteady compressible flows

DOCTORAL DISSERTATION OF:
LUCA CIRROTTOLA

Supervisor:
Prof. Giuseppe Quaranta
Co-Supervisor:
Dr. Barbara Re
Tutor:
Prof. Alessandro Airoldi
The Chair of the Doctoral Programme:
Prof. Pierangelo Masarati

2018 —Cycle XXX

Abstract

An interpolation-free conservative solution transfer methodology for body-fitted dynamic adaptive meshes is considered for the coupled numerical simulation of unsteady compressible flows and rigid body dynamics. The aim is to enable conservative mesh adaptation in three-dimensional transonic aeroelastic simulations with large relative free body motions, where element connectivity change becomes necessary to preserve the mesh quality, and an explicit solution interpolation from the old to the new mesh is known to reduce the solution accuracy. To this end, a continuous time interpretation of each local remeshing operation is employed to preserve solution conservation among different adapted meshes, avoiding any explicit interpolation step. A first application to a three-dimensional aeroelastic problem is presented with the analysis of the nonclassical aileron buzz. The methodology is validated by reproducing the self-sustained aileron oscillations in the flight speed and frequency range found in the experiments reported in literature. The large oscillations of the finite-span aileron require frequent remeshing, which is managed by the conservative methodology while also adapting the mesh to the shock waves pattern. A simple numerical methodology for shape interpolation is also introduced for the dynamic geometrical modeling of continuous wing–aileron configurations in two dimensions, and used for comparisons with discontinuous configurations for a preliminary assessment of the geometry effects on the aeroelastic phenomenon. Some complementary problems in mesh mechanics and structural mechanics are also individually addressed, with the development of a constant-connectivity mesh adaptation model and low-dimensional structural models for wing morphing.

Keywords. Conservative mesh adaptation, unsteady compressible flows, rigid fluid–structure interaction, nonclassical aileron buzz.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank all the people who have materially or morally contributed to the success of this work. It would not have been possible without the dedicated efforts and guidance of (in order of appearance) my supervisor Giuseppe Quaranta, Marco Morandini, Alberto Guardone, Barbara Re, Cécile Dobrzynski (a big, special thank you), Algiane Froehly, Mario Ricchiuto.

During these years I have met wonderful colleagues in both Italy and France. I would like to thank once again the patience and support of my parents and my friends.

Summary

This work presents a conservative methodology for unstructured mesh adaptation in the aeroelastic simulation of unsteady compressible flows over rigidly moving bodies and morphing boundaries. The objective is to enable a first application of conservative mesh adaptation to three-dimensional aeroelastic problems where topology-changing mesh adaptation is necessary, as in the case of fluid flow simulations using body-fitted meshes over bodies with large relative motion, and solution conservation is desirable, as in the case when shock wave motion influences aeroelastic stability. The conservative solution transfer methodology over adapted meshes relies on a continuous time interpretation of local remeshing operations. The volume swept by moving cell interfaces is computed for each transformation of mesh cells, so that conservation can be enforced in the solution transfer procedure without any explicit interpolation step.

The `Flowmesh` solver for unsteady compressible fluid flows over dynamic adaptive meshes, which originally implements the conservative mesh adaptation methodology applied in this work through the link with the `MMG` remeshing library, has been employed and extended to fluid–structure interaction problems. Algorithmic efforts have been devoted to the optimization of this fluid flow solver in order to enable its application to complex three-dimensional aeroelastic problems, and to the coupling with numerical procedures for structural interface displacement. The conservative methodology employed in the fluid flow solver also allowed a straightforward coupling with constant-connectivity mesh adaptation procedures for unsteady compressible flows, which have been investigated in their mesh mechanics aspects with the development of an original numerical model for mesh motion in the `FMG` library at the INRIA research institute in Bordeaux.

The subject of *morphing*, i.e. bodies capable of continuously changing their shape in order to adapt to given dynamic requirements, has also been addressed. From the computational fluid dynamics perspective, the interaction between mesh adaptation and changing boundary geometry has been targeted through the development of a shape interpolation procedure for the dynamic geometrical modeling of morphing boundaries. The procedure relies on standard geometric

models to build a dynamic parameterization capable of interacting flawlessly with mesh adaptation, allowing to position boundary mesh points on a moving curve with given geometrical accuracy while preserving solution conservation. From the structural mechanics perspective, a low-order finite element model for three-dimensional camber-morphing wings has been developed. This model is an extension of generalized beam models to camber-morphing wings — now restricted to constant cross-section wings — allowing to retain into the model only a desired number of degrees-of-freedom related to the in-plane deformation of the wing cross-section, together with the classical beam deformation modes. Spectral convergence of the reduced-order model to the full-order one has been proved in two dimensions, while the model has shown a convergent behavior also in three dimensions. The preliminary coupling of this structural model with a fluid flow solver through a conservative meshless fluid-structure interface scheme has also been addressed, highlighting the lack of consistency of the meshless interface scheme over non-matching discretizations.

Finally, a first three-dimensional aeroelastic application is presented with the numerical simulation of the nonclassical aileron buzz, where both conservative mesh adaptation and rigid fluid-structure interaction are applied. The analysis is validated by reproducing the self-sustained aileron oscillations in the flight speed range reported in the literature, and by comparing the computed oscillation frequency with measurements and simulations available from the literature. The shape interpolation procedure is used for two-dimensional comparisons of configurations with and without structural continuity between wing and aileron. These simulations, in addition to being a first application of conservative mesh adaptation to a three-dimensional aeroelastic problem with free body motion, allow to investigate the changes in the buzz phenomenology due to the three-dimensional geometry, and to highlight the effects of the shape of the wing-aileron connection on shock wave motion.

1 Introduction

1.1 Background and motivation

CFD on unstructured adaptive meshes. Computational Fluid Dynamics (CFD) is nowadays a mature technology with wide application both in research and industry, as testified by the large number of existing research and commercial simulation software packages. The current capabilities and limitations of this technology have been addressed, for example, in the recent NASA CFD Vision 2030 Study [150], where critical topics requiring research efforts in the next decade are also identified. Among them, mesh generation and adaptation have been highlighted as current bottlenecks in the CFD workflow, with several issues to be addressed to increase the robustness and automation in these steps. When dealing with complex geometries, unstructured meshes have shown greater versatility in producing body-fitted meshes, i.e. meshes conforming with solid body shapes. Unstructured mesh adaptation, i.e. adapting the mesh to the flow solution by changing elements shape and/or connectivity, holds the potential to control the discretization error by using the flow solution to drive mesh refinement and coarsening, while maintaining a desired level of refinement on complex boundaries. The current status of unstructured mesh adaptation can be found in recent reviews [129, 9], where strengths and limitations are presented together with a detailed historical perspective on the scientific contributions in this field, while an overview of current software packages implementing state-of-the-art techniques can be found in [9, 92]. The versatility of unstructured CFD methods is at the root of their increasing usage also in the field of aeroelasticity [21], where they are particularly suited for transonic flows on geometrically complex shapes, which are typically outside the domain of application of classical linear aeroelastic methods.

Conservative mesh adaptation and aeroelasticity. In the field of unstructured mesh adaptation, a critical issue in dynamic mesh adaptation, i.e. mesh adaptation performed at multiple time steps during an unsteady simulation, is the topic of solution transfer from the old to the adapted mesh. As shown in [5], generic interpolation algorithms can bring errors in the conservation of the solution (which

means errors in the conservation of mass, momentum, and energy for a finite-volume solver), which are accumulated in time during an unsteady flow simulation. It is worth mentioning that solution interpolation is an issue also shared by CFD methods other than unstructured adaptation of body-fitted meshes, namely overset grid methods and immersed boundary methods. Currently, conservative solution transfer algorithms have been developed for unstructured mesh adaptation. The most recent and active contributions to the topic can be found in [94, 7]. The latest works (appeared in 2017) on conservative mesh adaptation in the numerical solution of unsteady inviscid compressible flows show the successful application of conservative solution transfer procedures to three-dimensional problems with time-dependent boundary motion, both using mesh intersections [16] and swept volume [140] approaches. The first aim of the research work discussed in this thesis is to present a three-dimensional application of conservative mesh adaptation to an aeroelastic problem, where the boundary motion is also unknown.

The remainder of this chapter presents the context and background for the computational fluid dynamics and mesh mechanics tools that will be used in the fluid flow solver and the applications shown in this thesis. The aim is to introduce most of the mathematical and numerical concepts concerning the peculiarities of computational fluid dynamics on moving domains over unstructured adaptive meshes: Arbitrary Lagrangian–Eulerian (ALE) formulations for compressible fluid flows (section 1.2), the Geometric Conservation Law (section 1.3), the state of the art of unstructured mesh adaptation (section 1.4), and conservation solution interpolation among adapted meshes (section 1.5). Finally, an introduction on the constant-connectivity and topology-changing mesh adaptation frameworks used in this work is given in sections 1.6 and 1.7, respectively, while the thesis outline is given in section 1.8.

1.2 Arbitrary Lagrangian–Eulerian formulation of compressible fluid flow

We are considering a subset of fluid flow problems of aeronautical interest, namely compressible (mostly transonic) flows around moving airfoils, wings, and helicopter blades. To this end, we look for a numerical solution of a conservative (finite volume) formulation of the Euler equations for compressible fluid flows [11, 109] on dynamically adaptive unstructured meshes. The need for moving meshes stems from the need of maintaining a body-fitted mesh throughout the time simulation with moving boundaries, and the Arbitrary Lagrangian–Eulerian framework provides a well-assessed method to handle both the boundary and the mesh movement [46].

We start from the conservative form of the Euler equations on a fixed, Eulerian domain Ω

$$\int_\Omega \frac{\partial \mathbf{u}}{\partial t}\,\mathrm{d}\Omega + \oint_{\partial\Omega} \hat{\mathbf{n}}\cdot\mathbb{F}(\mathbf{u})\,\mathrm{d}\Gamma = \mathbf{0} \tag{1.1}$$

where $\mathbf{u}$ is the array of the conservative solution, composed by the mass density ρ, the momentum density $\rho\mathbf{U}$ and the total energy density ρe^t, and $\mathbb{F}(\mathbf{u})$ is its flux

$$\mathbf{u} = \begin{bmatrix} \rho \\ \rho\mathbf{U} \\ \rho e^t \end{bmatrix}, \qquad \mathbb{F}(\mathbf{u}) = \begin{bmatrix} \rho\mathbf{U} \\ \rho\mathbf{U}\otimes\mathbf{U} + P(\rho,e)\mathbb{I} \\ \rho e^t\mathbf{U} + P(\rho,e)\mathbf{U} \end{bmatrix} \tag{1.2}$$

Pressure $P(\rho,e)$ is computed through a suitable equation of state.

Equations 1.1 are valid on each element Ω_k in a triangulation of the domain Ω. We are considering an arbitrary moving domain $\Omega(t)$, not necessarily a material domain (from which the name *Arbitrary Lagrangian–Eulerian*). This means both allowing the domain to move following the motion of the boundary $\partial\Omega(t)$ (as in the case of aeroelastic simulations), but also allowing the movement of mesh elements $\Omega_k(t)$ while keeping the boundaries fixed, as it can be the case in mesh adaptation (see section 1.5 and chapter 2).

Let $\mathbf{X}\in\Omega$ be a parameterization of the fixed domain, and $\mathbf{x}\in\Omega(t)$ be a parameterizarion of the moving domain. The coordinates of the moving domain can be expressed as a function of the fixed domain coordinates $\mathbf{X}$ and time t

$$\mathbf{x} = \boldsymbol{\phi}(\mathbf{X}, t) \tag{1.3}$$

so that a generic tensor field $\mathbf{f}(\mathbf{x},t)$ over the moving domain $\Omega(t)$ can be expressed as

$$\mathbf{f}(\boldsymbol{\phi}(\mathbf{X},t),t) = \mathbf{f}(\mathbf{X},t) \tag{1.4}$$

With an abuse of notation, the functional dependence is used to denote the domain over which the field $\mathbf{f}$ is defined. From the last relation, the chain rule can be applied in order to define derivatives of $\mathbf{f}$, for example

$$\left.\frac{\partial \mathbf{f}(\mathbf{x},t)}{\partial t}\right|_{\mathbf{X}} = \left.\frac{\partial \mathbf{f}}{\partial t}\right|_{\mathbf{x}} + \left.\frac{\partial \boldsymbol{\phi}}{\partial t}\right|_{\mathbf{X}}\cdot\nabla_{\mathbf{x}}\mathbf{f} \tag{1.5}$$

The velocity of the moving domain, which is the mesh velocity relative to the fixed domain, is thus defined as

$$\mathbf{v} \triangleq \left.\frac{\partial \boldsymbol{\phi}}{\partial t}\right|_{\mathbf{X}} \tag{1.6}$$

Since we are interested in conservative formulations, it is interesting to consider the Reynolds' transport theorem for a generic tensor field $\mathbf{f}$

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)} \mathbf{f}\,\mathrm{d}\Omega = \int_{\Omega(t)} \frac{\partial \mathbf{f}}{\partial t}\,\mathrm{d}\Omega + \oint_{\partial\Omega(t)} \hat{\mathbf{n}}\cdot\mathbf{v}\mathbf{f}\,\mathrm{d}\Gamma \tag{1.7}$$

By applying the Reynolds' transport theorem to the conservative equations in Eulerian form (eq. 1.1), we obtain the ALE form of the conservative equations

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)} \mathbf{u}\,\mathrm{d}\Omega + \oint_{\partial\Omega(t)} \hat{\mathbf{n}}\cdot(\mathbb{F}(\mathbf{u}) - \mathbf{v}\mathbf{u})\,\mathrm{d}\Gamma = \mathbf{0} \tag{1.8}$$

The domain motion taken into account by velocity $\mathbf{v}$ in eq. 1.8 can be originated by the motion of a solid boundary, which is usually propagated into the domain by means of an elastic analogy [95]. If a constant-connectivity mesh adaptation methodology is considered (see section 1.6), this also can be straightforwardly seen as a domain motion amenable to be taken into account through velocity $\mathbf{v}$ in the ALE equations 1.8. Topology-changing mesh adaptation can also be taken into consideration in the ALE equations once it is provided with a continuous time interpretation. This is done in the conservative solution transfer methodology used in this work, which will be introduced in section 1.5 and explained in further depth in section 2.

1.3 Geometric Conservation Law

Altought still a debated topic, the so-called Geometric Conservation Law (GCL) is a generally accepted [55] consistency requirement between the mesh elements volume change in time and the mesh velocities

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)} \mathrm{d}\Omega = \oint_{\partial\Omega(t)} \hat{\mathbf{n}}\cdot\mathbf{v}\,\mathrm{d}\Gamma \tag{1.9}$$

This relation was first introduced in [161] and further discussed in [156] as an additional conservation law to be satisfied by numerical simulations of unsteady flows on moving domains. The importance and necessity of fulfilling a discrete Geometric Conservation Law (DGCL), specific for each discretization scheme, in numerical simulations on moving domains is subject to a vivid debate, which is well summarized in the detailed review given in [55].

Time accuracy and *numerical stability* are the main issues related with the analysis of the GCL. Several works aimed at providing proofs of sufficiency and necessity of a DGCL for time-accuracy and numerical stability for selected ALE schemes. In [78] it is proved that the DGCL is a sufficient condition for a numerical scheme which is p-th order time-accurate on a fixed grid to be at least first-order accurate on moving grids, while in a following paper [71] some time-accurate numerical schemes not compliant with a DGCL are designed, proving that the DGCL is not a necessary condition for reaching the design order of accuracy in time, when the latter is higher than one. Design of integration schemes that are both high-order time-accurate and DGCL-compliant is presented in [117]. In [57] it is

proved for sample ALE schemes that fulfilling a DGCL is a necessary and sufficient condition for a numerical scheme which is nonlinearly stable on fixed grids to retain nonlinear stability on movin grids, while in [28] opposite results are shown for parabolic problems.

Even if definitive statements of sufficiency or necessity of a DGCL for the time accuracy or numerical stability of general ALE schemes are missing, there are evidences that not fulfilling a discrete version of the Geometric Conservation Law (DGCL) determines the onset of spurious oscillations in the numerical solution [57, 115], thus the GCL is indeed required for *numerical consistency*. In fact, even if it can be derived from geometric considerations (in a procedure similar to the proof of Reynolds' transport theorem), the GCL is ofter reported as a condition on the reproduction of a uniform flow field by the numerical scheme. Of course, the purpose is not the simulation of a uniform flow field per se; instead, the constant solution is the easiest consistency test that can be performed in numerical analysis, and the uniform flow field test is similar to the *patch test* in finite element methods [127]. It is easy to show that given an uniform flow field $\mathbf{u}(\mathbf{x}, t) \equiv \mathbf{w}(t)$ on an infinite domain, according to the conservative Euler equations of gas dynamics it should satisfy the relation

$$|\Omega| \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = \mathbf{0} \tag{1.10}$$

on any bounded fixed domain Ω, thus the field $\mathbf{w}$ is also constant in time. Plugging this solution into the ALE formulation of the governing equation on a moving domain $\Omega(t)$, we get

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(|\Omega|\mathbf{w}\right) + \oint_{\partial\Omega(t)} \hat{\mathbf{n}} \cdot \mathbf{v} \mathrm{d}\Gamma \mathbf{w} = \mathbf{0} \tag{1.11}$$

so

$$|\Omega(t)| \frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} + \left(\frac{\mathrm{d}|\Omega|}{\mathrm{d}t} + \oint_{\partial\Omega(t)} \hat{\mathbf{n}} \cdot \mathbf{v} \mathrm{d}\Gamma \right) \mathbf{w} = \mathbf{0} \tag{1.12}$$

From the last relation, we see the uniform flow field cannot remain constant in time if the GCL is not satisfied, and this gives rise to a zeroth-order, consistency error in the satisfaction of eq. 1.10.

Beside the theoretical discussion and the tangible implications on the solution consistency, from the practical point of view the GCL provides an additional constraint on the flow equations that is reflected on the algorithm chosen for the computation of mesh velocities needed in ALE formulations (see eq. 1.12). Thus, several works have focused on designing algorithms for the evaluation of mesh velocity (typically their component normal to element faces in finite volume schemes, or their gradient and divergence in finite element schemes) while automatically fulfilling a DGCL condition [117, 55, 94].

1.4 State of the art of unstructured mesh adaptation

Unstructured mesh adaptation is becoming an integral part of the Computational Fluid Dynamics (CFD) simulation workflow [129], together with the initial mesh generation, the numerical solution of the flow equations — possibly optimized for a High Performance Computing (HPC) environment — and a postprocessing phase. In an unsteady flow simulation over moving boundaries, mesh adaptation can accomplish a twofold objective:

1. *Locally refining/coarsening the mesh* according to the flow solution, in order to improve the solution accuracy without the computational overhead that would be caused by a uniform mesh refinement strategy.

2. *Modeling moving boundaries* during unsteady simulations, as in the case of fluid–structure interaction and aeroelastic simulations, while maintaining a body-fitted mesh.

Mesh refinement/coarsening needs the development of an indicator function to guide the mesh adaptation phase. This indicator function depends on the flow solution, and its definition determines the type of mesh adaptation that is being performed:

- *Feature-based* and *Hessian-based* mesh adaptation, where the indicator function is designed to target specific flow pattterns through a convenient combination of the solution and its derivatives (typically the gradient and Hessian) or to control the interpolation error by means of the reconstructed Hessian of the solution [9]. Some issues with this type of indicators are well-known (gradients can possibly follow shocks in the wrong location if a too coarse initial grid is used [129, 166], and the reconstructed Hessian can be non-convergent [129, 98]), but they have been successfully used for engineering adaptation purposes.

- *Goal-oriented* mesh adaptation, where the indicator function is designed to minimize the error on a specified functional of the flow solution (such as lift or drag). This indicator is typically an adjoint-weighted error estimator which requires an adjoint flow solution to be computed [61, 9].

It is worth mentioning that the aim of **modelling moving boundaries** can be currently tackled with other methods alternative to the body-fitted meshes. For example:

- In *overset grid methods* [29, 81], the computational domain is split in overlapping blocks, each of them meshed independently in order to ease the

problem of generating a body-fitted mesh around complex geometries. Continuity among the different blocks is then restored by means of interpolation conditions that the solution is required to satisfy on the overlapping zones of the domain. As explained in the next sections, with these methods it could be difficult to ensure conservation [165].

- In *immersed boundary methods* [123] a body-fitted mesh is never produced, and the fluid flow equations are solved on a geometrically simpler, wider computational domain that includes the boundary in its interior. Boundary conditions are reproduced by means of a specifically designed artificial forcing term which is introduced into the flow equations [123], or by means of cut-cell methods [151]. Difficulties with this class of methods usually arise when trying to improve grid resolution near solid walls. The combination of immersed boundary methods with unstructured mesh adaptation for boundary layers has been proposed in several works [151, 1].

Although a thorough analysis of these methods is beyond the scope of this work, we note that they are not free from conservation issues as well. Details about conservation problems arising from solution interpolation in overset grid methods are presented in [165], while a discussion on the lack of conservation near sharp immersed boundaries can be found in [147].

The specific mechanisms of unstructured mesh adaptation vary depending on whether mesh connectivity has to be preserved, or not. Common strategies for constant-connectivity mesh adaptation (also called moving mesh methods) consider recasting the problem in terms of partial differential equations (driven by the indicator function as a forcing, or variable stiffness term), so they are global methods. Topology-changing mesh adaptation, instead, relies on a set of mesh modifications (like node insertion, removal, edge swap and element split) that can be performed locally whenever a certain threshold in the indicator function is reached.

1.5 Conservative interpolation

Solution reconstruction on adaptive meshes. Every time that some type of mesh adaptation is performed (being node relocation, edge swapping, node insertion or removal), the discrete solution values known on the previous mesh (commonly referred to as the *background* mesh [65, 7] need to be transferred to current mesh entities.

The solution reconstruction process can be conceptually split in two steps [7]:

1. *Localization* of current nodes into elements of the previous mesh (or vice versa, depending on the algorithm), in order to select a stencil of nodes to be used for the solution mapping from the background to the current mesh.

2. *Solution transfer* on the current mesh, given the solution on the background mesh.

Localization relies on search algorithms, so its computational cost depends on the underlying data structures used in the program at hand. A common algorithm consists in building a search path by going from neighbour to neighbour until the node is located into an element [65, 114, 6]. Typical difficulties with this algorithm can arise with non-convex domains, holes inside the domain and non-matching[1] surface discretizations (see [65, 114, 6] for a thorough discussion of localization algorithms). Since searching is particularly efficient on grid data structures and quadtrees/octrees, the above algorithm is frequently coupled with grid/octree searching to restrict the space region where the search path is built [114, 6]. Localization algorithms are typically unnecessary in conservative solution transfer methods which track local modification in time (as the one considered in this work), thus implicitly localizing new mesh nodes into the old mesh.

The solution transfer step allows to use the discrete solution available on the background mesh to reconstruct the current solution. For example, a classical linear interpolation of nodal values would need the localization of each current node $\mathbf{x}_i^{\text{new}}$ into an element Ω_k^{old} of the previous mesh, so that the current values can be computed from the evaluation of linear shape functions at the coordinates of the new node (so-called area coordinates)

$$\mathbf{u}(\mathbf{x}_i^{\text{new}}) = \sum_{j=i}^{d} \theta_j^{(k)}(\mathbf{x}_i^{\text{new}})\mathbf{u}(\mathbf{x}_j^{\text{old}}) \tag{1.13}$$

Again, conservative solution transfer methods which track local mesh modification in time are able to skip any explicit solution transfer step.

A number of properties contribute to the quality of a solution transfer algorithm:

- *Efficiency*, in terms of memory usage, computational cost (number of operations, global or local nature of the algorithm) and parallel scalability [114].

- *Accuracy*, especially when adaptation and solution transfer has to be performed several times during the time simulation [6].

[1] In this case, by *non-matching* surface discretizations we denote two different discretizations of the same geometric boundary, producing gaps and overlaps between the background and the current mesh.

- *Capability of handling of non-matching discrete boundaries*, since a non-matching geometry typically requires some ad-hoc modifications of the solution reconstruction algorithm [7].

Local solution conservation In numerical simulations based on a conservative formulation of the governing equations, standard solution interpolation procedures do not allow an automatic preservation of the conservation property. In unsteady compressible flow simulations, where adaptation is performed recurrently at many time steps, the interpolation error accumulates throughout time integration, and can indeed become the main source of numerical errors in the solution [5]. Outside the domain of mesh adaptation, the adverse effects of standard solution interpolation techniques has been recognized also in overset grids methods [165]. For this reason, several works have focused on the study of *conservative* interpolation procedures which guarantee, at least, the conservation of the volume integral of the solution (i.e. the L^1 norm, commonly referred as the *mass* of the solution) from the background to the current mesh

$$\int_{\Omega^{\text{old}}} \mathbf{u}^{\text{old}} \, \mathrm{d}\Omega = \int_{\Omega^{\text{new}}} \mathbf{u}^{\text{new}} \, \mathrm{d}\Omega \tag{1.14}$$

where the superscripts *old* and *new* mean that we are dealing with the numerical approximations of the integral on the old and new mesh, respectively.

It is worth noticing that the last equation should hold both globally and locally, and locality is often exploited to build the conservative interpolation algorithm [59, 7, 94, 140].

Some families of approaches are reported here, without the ambition of being complete. The main conceptual difference among them relies in attempting a direct numerical evaluation of eq. 1.14, or reconvering conservation by considering each element in the current mesh as an evolution of a parent element in the background mesh.

1. *Galerkin projection* — Several works [69, 59] enforce conservation by performing a Galerkin projection of the solution from the finite dimensional space $\mathscr{U}^{\text{old}}$ on the old mesh to the finite dimensional space $\mathscr{U}^{\text{new}}$ on the new mesh as

$$\langle v^{\text{new}}, u^{\text{new}} \rangle_{L^2(\Omega)} = \langle v^{\text{new}}, u^{\text{old}} \rangle_{L^2(\Omega)}, \quad \forall \, v^{\text{new}} \in \mathscr{U}^{\text{new}} \tag{1.15}$$

   The left hand side of eq. 1.15 can be computed element-wise on the new mesh Ω^{new}, while the right hand side needs the computation of the intersection of each elements Ω_k^{new} of the new mesh with the set of elements of the old mesh $\{\Omega_j^{\text{old}} \subset \Omega^{\text{old}} : \Omega_j^{\text{old}} \cap \Omega_k^{\text{new}} \neq \emptyset\}$ partially covering it, since the

right hand side of eq. 1.15 involves products of functions defined on the two different meshes. The same need for computing intersections arises in any direct numerical evaluation of eq. 1.14.

Alternatively, the right hand side of eq. 1.15 can be numerically evaluated by building a *supermesh* [60, 3], made of a constrained Delauney triangulation of the union of nodes and edges of both the background and current meshes [60]. In [59] the supermesh approach is applied locally rather than globally for the assembly of the right hand side.

The projection method produces a sparse linear system to be solved, but is still a global method.

2. *Local mesh intersections* — These methods enforce the conservation property locally on the intersection of each element of the new mesh Ω_k^{new} with the elements the old mesh Ω^{old}, possibly with the introduction of a supermesh [118]. Then, the solution is trasferred by means of a solution reconstruction procedure [7, 118].

3. *Swept volumes integration* — This family of approaches stems from the geometric consideration that an arbitrary variation of an infinitesimal element volume can be written as a function of the volumes swept by a variation $\delta\mathbf{x}$ in the position of its boundaries $\mathrm{d}\Gamma$

$$\delta\,\mathrm{d}\Omega_k = \hat{\mathbf{n}}\cdot\delta\mathbf{x}\,\mathrm{d}\Gamma \tag{1.16}$$

so the variation of a function $\mathbf{u}$ integrated over an element Ω_k can be split in a term related to the intrinsic variation of $\mathbf{u}$, and a term related to volumes swept by the boundaries of Ω_k

$$\delta\int_{\Omega_k(t)}\mathbf{u}\,\mathrm{d}\Omega = \int_{\Omega_k(t)}\delta\mathbf{u}\,\mathrm{d}\Omega + \oint_{\partial\Omega_k(t)}\hat{\mathbf{n}}\cdot\delta\mathbf{x}\mathbf{u}\,\mathrm{d}\Gamma \tag{1.17}$$

The last term in the right hand side represents the variation in the integral of the function field $\mathbf{u}$ due to volume variations only. This approach was introduced in [50] and has been used to enforce conservation in Arbitrary Lagrangian-Eulerian methods where solution and grid are firstly updated with a Lagrangian phase, then the nodes position is enhanced with a rezoning phase, and lastly the Lagrangian solution is interpolated on the rezoned grid with in a remapping phase [50, 116]. This method is naturally suited for constant-connectivity meshes, as it is always possible to apply this procedure at each mesh element Ω_k and to compute the volume swept by its faces. This approach has been extended to changing-connectivity meshes in [100], under the restriction of a single reconnection per edge, by means of

a three-steps procedure. Firstly, the deleted edge is shrunk into its barycenter; secondly, the new edge connectivity is established; lastly, the new edge is expanded until its final nodes configuration. This fictional splitting of the edge reconnection allows for the computation of swept volumes even with connectivity change.

More recently, an analogous time evolution interpretation of topology change for unstructured mesh adaptation has been introduced in [75, 94, 140] in order to preserve conservation and automatically fulfill a DGCL condition in unsteady ALE schemes where adaptation is performed between time step $t^{(n)}$ and $t^{(n+1)}$. This is the computational approach used in this work, and it will be further discussed in the following chapters. Instead of explicitly interpolating the solution from the background to the current mesh, the swept-volumes approach is used to numerically evaluate the mesh interface velocities $\mathbf{v}$ which are necessary for the solution of the governing equations in ALE form (eq. 1.12). In fact, the change in volume of each cell reads

$$\Delta|\Omega_k| = |\Omega_k|^{(n+1)} - |\Omega_k|^{(n)} = \int_{t^{(n)}}^{t^{(n+1)}} \oint_{\partial\Omega_k(t)} \hat{\mathbf{n}} \cdot \mathbf{v} \, \mathrm{d}\Gamma \, \mathrm{d}t \tag{1.18}$$

and it can be further split into the contribution given by each cell interface $\Gamma_{ki}(t)$ as

$$\Delta|\Omega_k| = \sum_{i=1}^{N_i} \Delta|\Omega_{ki}| \tag{1.19}$$

where

$$\Delta|\Omega_{ki}| = \int_{t^{(n)}}^{t^{(n+1)}} \int_{\Gamma_{ki}(t)} \hat{\mathbf{n}} \cdot \mathbf{v} \, \mathrm{d}\Gamma \, \mathrm{d}t \tag{1.20}$$

Since the volumes swept by each cell interface can be computed also with changing connectivities thanks to the three-steps procedure previously outlined, coupling of the DGCL condition (eq. 1.20) with the ALE formulation of the governing equations (eq. 1.12) provides a closed system to be solved for the flow solution $\mathbf{u}$ and the cell interface velocities $\mathbf{v}$.

Finally, it is worth noting that a similar time evolution interpretation of local mesh adaptation has been proposed in [101], where it is exploited for the construction of space-time elements compatible with mesh adaptation in order to preserve conservation of linear and angular momentum in Lagrangian dynamics formulations of continuum mechanics.

Aside from a comparison of the accuracy of each scheme, the point of view of the implementation in a computer code also deserves some attention. A straigthforward advantage of swept-volumes approaches over the mesh-intersection ones

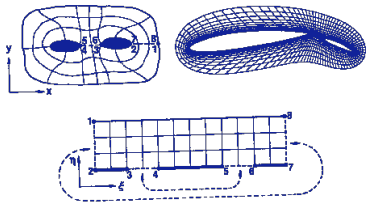


Figure 1.1 – Structured grid generation around solid bodies (from [157]).

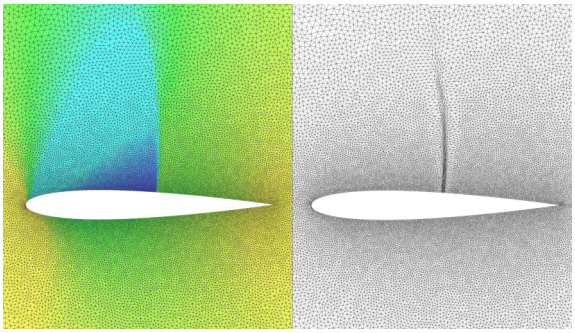


Figure 1.2 – Laplacian-based r-adaptation on a steady compressible flow solution. Left: Initial grid and density solution. Right: Adapted mesh.

Figure 1.3 – Laplacian-based grid generation and mesh adaptation.

is that the first ones do not need search algorithms to localize new mesh nodes into the background mesh. However, this comes at the cost of tracking each local mesh modification for the computation of the swept volumes. This requires the implementation of data structures well-suited for mesh adaptation [63], and the development of a tight coupling of the flow solver with the remesher software.

1.6 Constant-connectivity mesh adaptation

Mesh adaptation with constant elements connectivity, based only on nodes *relocation* (from which the name *r-adaptivity*, or *moving mesh methods*) offers the interesting possibility of preserving the one-to-one node mapping from the old to the new mesh. This is particularly useful in Arbitrary Lagrangian–Eulerian formulations of the flow equations, since by providing a GCL-compliant numerical evaluation of mesh velocities $\mathbf{v}$ it is possible to avoid any solution interpolation step, as the solution on the new mesh will be updated through the flow equations in ALE form (eq. 1.12). Following the review given by Budd et al. in [31], moving mesh methods can be classified as **velocity-based** or **location-based**, depending on whether a solution for the velocity or the position of the mesh nodes is sought. The former has evolved from Lagrangian formulations of fluid dynamics, where the position of mesh nodes is obtained through time integration of particle velocity. The latter can be further subdivided in two main families of approaches.

- Historically, the origins of **Laplacian-based** mesh generation methods can be linked to the analogy with potential flow theory. Through the solution of a Laplace equation $\nabla^2\phi = 0$, smooth flow potential isolines $\phi = \text{const}$ are produced around curved bodies, like the curvilinear coordinate system one

would like to achieve in body-fitted structured grid generation (as in figure 1.1). If we define the coordinates $\mathbf{x} = (x_1, \dots, x_d)$ on the physical domain $\Omega_{\mathbf{x}} \in \mathbb{R}^d$ and the curvilinear coordinates $\boldsymbol{\xi} = (\xi_1, \dots, \xi_d)$ on a reference domain $\Omega_{\boldsymbol{\xi}} \mathbb{R}^d$, a body-fitted equidistribution of the reference coordinates $\boldsymbol{\xi}$ could than be obtained by solving the Laplace equation[2]

$$\nabla^2_{\mathbf{x}} \boldsymbol{\xi} = \mathbf{0} \tag{1.21}$$

in the physical domain $\Omega_{\mathbf{x}}$ (this implies an independent Laplace equation $\nabla^2_{\mathbf{x}} \xi_i = 0$ in each space direction $i = 1, \dots, d$). This is a classical approach in structured body-fitted grid generation [157], which can be then generalized to a Poisson equation

$$\nabla^2_{\mathbf{x}} \boldsymbol{\xi} = \mathbf{p}(\mathbf{x}) \tag{1.22}$$

in order to drive mesh points in specific zones through the source term $\mathbf{p}(\mathbf{x})$, or through a variable diffusion equation introduced by Winslow [169]

$$\boldsymbol{\nabla}_{\mathbf{x}} \cdot (D(\mathbf{x}) \boldsymbol{\nabla}_{\mathbf{x}} \boldsymbol{\xi}) = \mathbf{0} \tag{1.23}$$

where the diffusion coefficient $D(\mathbf{x})$ allows to gather mesh points in specified zones, while providing an automatic stiffening effect in elements with shrinking dimension. Equations 1.21, 1.22, and 1.23 are usually transformed from the physical domain $\Omega_{\mathbf{x}}$ to the reference one $\Omega_{\boldsymbol{\xi}}$ in order to directly solve them for the mesh point coordinates $\mathbf{x}(\boldsymbol{\xi})$. After the transformation, the resulting PDEs are nonlinear, but the reference domain $\Omega_{\boldsymbol{\xi}}$ is generally chosen as a combination of rectangular subdomains [157], making it amenable to standard finite difference methods [169]. Passage from mesh generation to mesh adaptation is possible by replacing the source or diffusion terms with a monitor function ω dependent from the fluid flow solution and its derivatives. Extension to unstructured meshes and finite element solutions has been straightforward, once this approach has been generalized to include elliptic PDEs obtained from the minimization of an adaptation functional (as in [87, 91, 51], among others). Dynamic mesh adaptation can be performed either by solving a steady elliptic problem for the mesh adaptation at each time step (as in [154, 38]) or by adding the time dimension to the adaptation functional (for example in [111, 90]). Alternatively, this approach can be used to formulate elliptic problems directly in the reference domain, as proposed in [37] and further extended in [153] to consider its application to hyperbolic conservation laws, in [38] to compressible multicomponent flows on triangular meshes, in [13, 14] to the shallow

[2] The notation $\boldsymbol{\nabla}_{\mathbf{x}}$, $\nabla^2_{\mathbf{x}}$ is used here to explicitly remark that derivation is performed in the physical domain $\Omega_{\mathbf{x}}$.

water equations. To the best of the author knowledge, not many applications to three-dimensional meshes are available to date, being [110] one of them. More recently, a geometric approach [32] to the solution of the mesh PDE has been introduced in [88], and in [89] it has shown that the geometric approach can be successfully used to preserve mesh nonsingularity during adaptation.

- **Elasticity-based** methods consider mesh deformation as the elastic deformation of the computational domain $\Omega_{\mathbf{x}}$, considered as a fictitious elastic solid, so they typically employ a Lagrangian formulation of the linear elasticity equations in the reference domain $\Omega_{\boldsymbol{\xi}}$. This approach is particularly used with moving boundaries and interfaces [95, 96, 152, 97]. In order to avoid mesh tangling for large mesh movements, various solutions have been proposed in order to introduce variable stiffness dependent from the mesh element size, such as multiplying the stiffness terms by the Jacobian of the coordinate transformation [95] or using an elastic modulus dependent on the undeformed element size [94]. In [128], the variable stiffness approach is used to drive mesh adaptation by modifying the Lamé coefficients of linear elasticity through the gradient of a monitor function based on the flow solution, inspired by the Laplacian-based works.

Arbitrary Lagrangian–Eulerian formulations of the fluid flow equations automatically provide a conservative solution remapping with constant-connectivity meshes.

1.7 Topology-changing mesh adaptation

In unstructured mesh adaptation with connectivity change and node insertion or removal, local remeshing has proven to be the most efficient strategy over a global remeshing approach [129]. Classical local mesh adaptation operations include nodes relocation, edge split, edge collapse, edge swap, and barycentric element split. These operations are applied iteratively to improve the mesh according to a provided indicator function. Usually, the indicator function is an error indicator that is used to mark elements for refinement or coarsening, according to given thresholds[167]. Alternatively, the error indicator can be used to specify a target mesh size.

The idea of specifying a target edge size map has been generalized in anisotropic mesh adaptation, where there is also the need for specifyng desired directions for element size growth. In the so-called **metric-based** framework for unstructured mesh adaptation [129, 66, 8, 124], this is accomplished by interpreting

the error indicator as a curved Riemaniann surface. Distance and orientation information become thus encoded into the definition of the metric tensor that is associated to the surface, giving more weight to highly curved regions of space [105]. The idea of looking for a transformation $\mathbf{x}(\boldsymbol{\xi}) : \Omega_\xi \rightarrow \Omega_\mathbf{x}$ between a reference domain Ω_ξ and the computational domain $\Omega_\mathbf{x}$ is recurrent and has been exploited in various ways in mesh generation [157, 65], error equidistribution in constant-connectivity mesh adaptation [51] and of course in ALE methods [46]. In metric-based mesh adaptation, the scalar product used for distance and volume computations is modified according to a specified metric map given by a tensor field $\mathcal{M}(\mathbf{x})$, so that generating a uniform mesh made of unit-length edges in the metric $\mathcal{M}$ (a so-called *unit-mesh*) automatically produces gathering and orientation of new elements according to the specific patterns, given by eigenvectors of the metric field $\mathcal{M}$. The scalar product between two vectors $\mathbf{a}(\mathbf{x})$ and $\mathbf{b}(\mathbf{x})$ then reads

$$(\mathbf{a},\mathbf{b})_\mathcal{M} = \mathbf{a}(\mathbf{x}) \cdot \mathcal{M}(\mathbf{x}) \cdot \mathbf{b}(\mathbf{x}) \tag{1.24}$$

and consequently the norm becomes

$$||\mathbf{a}||_\mathcal{M} = \sqrt{\mathbf{a} \cdot \mathcal{M}(\mathbf{x}) \cdot \mathbf{a}} \tag{1.25}$$

Given this notion of edge length, the prescription of the metric field can be linked to a posteriori error estimation, where some bounds for the approximation error $||u - u_h||$ is sought. This is provided by Céa's lemma [138], which states that the approximation error is bounded by the interpolation error $||u - \Pi_h u||$ (where Π_h is a Lagrange interpolant)

$$||u - u_h|| \leq c||u - \Pi_h u|| \tag{1.26}$$

Even if this result is strictly valid only for elliptic problems, it has proven to be experimentally valid even for hyperbolic problems [8, 66]. Thanks to a Taylor's expansion of the interpolation error, the following estimate holds [66]

$$||u - \Pi_h u||_{\infty,K} \leq c_d \max_{\mathbf{x}\in K} \max_{\mathbf{e}\in E_K} \mathbf{e} \cdot |\mathcal{H}_u(\mathbf{x})| \cdot \mathbf{e} \tag{1.27}$$

Where $\mathcal{H}_u(\mathbf{x})$ is the Hessian of the solution, and c_d is a constant related to the space dimension. Following [66], the following estimate holds

$$||u - \Pi_h u||_{\infty,K} \leq c_d \max_{\mathbf{e}\in E_K} \mathbf{e} \cdot \mathcal{M}(\mathbf{x}) \cdot \mathbf{e} \tag{1.28}$$

where the desired metric tensor map $\mathcal{M}(\mathbf{x})$ is defined as a corrected eigendecomposition of the Hessian tensor of the solution

$$\tilde{\mathcal{M}} = \mathbf{R}\tilde{\mathbf{\Lambda}}\mathbf{R}^{-1}, \qquad \tilde{\Lambda} = \begin{bmatrix} \tilde{\lambda}_1 & 0 & 0 \\ 0 & \tilde{\lambda}_2 & 0 \\ 0 & 0 & \tilde{\lambda}_3 \end{bmatrix} \tag{1.29}$$

$$\tilde{\lambda}_i = \min\left(\max\left(\frac{c_d|\lambda_i|}{\epsilon}, \frac{1}{h_{\max}^2}\right), \frac{1}{h_{\min}^2}\right) \tag{1.30}$$

with $\lambda_i, \mathbf{R}$ being the eigenvalues and eigenvectors of the Hessian tensor, $h_{\min}, h_{\max}$ being the minimum and maximum desired edge length, and ϵ being the maximum tolerated interpolation error.

This error estimation has been deemed geometric by its authors, in the sense that it can be interpreted as the difference between the cartesian surface defined by the solution $u(\mathbf{x})$ and the piecewise linear approximation of the same surface $\Pi_h u$. In this sense, this error estimator should be independent from the specific PDE problem at hand.

This framework automatically includes isotropic mesh adaptation, which is recovered by assigning an isotropic metric tensor

$$\mathcal{M}(\mathbf{x}) = \begin{bmatrix} \tilde{\lambda}_{\text{iso}}(\mathbf{x}) & 0 & 0 \\ 0 & \tilde{\lambda}_{\text{iso}}(\mathbf{x}) & 0 \\ 0 & 0 & \tilde{\lambda}_{\text{iso}}(\mathbf{x}) \end{bmatrix} \tag{1.31}$$

$$\tilde{\lambda}_{\text{iso}}(\mathbf{x}) = \min\left(\max\left(\frac{1}{h_{\text{iso}}^2(\mathbf{x})}, \frac{1}{h_{\max}^2}\right), \frac{1}{h_{\min}^2}\right) \tag{1.32}$$

In isotropic mesh adaptation, the isotropic size map $h_{\text{iso}}(\mathbf{x})$ is often built relatively to the previous mesh size, using the error indicator to provide thresholds for refinement or coarsening. For smooth problems, interpolation estimates based on linear functions propose the Hessian of the solution as a natural candidate for error estimation [167, 114], and the Hessian indicator has been extended to avoid singularities when dealing with non-smooth features such as shocks [167, 94]. After an error estimator is computed, this is used to mark the elements to be targeted for refinement/coarsening according to given thresholds, possibly performing multiple hierarchical evaluations [2].

1.8 Thesis contributions and manuscript organization

The first aim of the research work discussed in this thesis is to extend an in-house conservative solution transfer technique, developed at the Department of Aerospace Science and Technology of Politecnico di Milano, to a three-dimensional aeroelastic application. Original contributions presented in this work concern:

C1: The aeroelastic numerical simulation of three-dimensional nonclassical aileron buzz with conservative topology-changing mesh adaptation.

C2: The development of a simple shape interpolation procedure for the simulation of two-dimensional compressible flows over morphing boundaries with conservative topology-changing mesh adaptation.

C3: The development of a numerical model for dynamic constant-connectivity mesh adaptation.

C4: The development of a low-dimensional structural model for three-dimensional constant-section camber-morphing wings.

Results have been reported in the following international conferences:

- Proceedings:
  - L. Cirrottola, M. Morandini, G. Quaranta, *A generalized beam formulation for the dynamic analysis of camber-morphing helicopter blades*. 2015 International Forum on Aeroelasticity and Structural Dynamics (IFASD), St. Petersburg, Russia, June 28–July 2, 2015.
  - L. Cirrottola, G. Quaranta, B. Re, C. Dobrzynski, A. Guardone, *Numerical simulation of nonclassical aileron buzz over 3D unstructured adaptive meshes*, ECCOMAS ECCM–ECFD 2018, Glasgow, June 11–15, 2017.
  - L. Cirrottola, M. Morandini, G. Quaranta, *Generalized beam models analysis for aeroelastic morphing applications*, ECCOMAS ECCM–ECFD 2018, Glasgow, June 11–15, 2017.
- Conferences:
  - L. Cirrottola, B. Re, G. Quaranta, *Simulation of compressible flows over opening wing-flap configurations*, International Conference on Adaptive Modeling and Simulation (ADMOS), Verbania, June 25–28, 2017.

This manuscript is structured as follows.

- **Part I** presents the conservative methodology for unsteady compressible flows over adaptive meshes with topology change (chapter 2) and adaptive meshes with constant connectivity (chapter 3, **C3**).
- **Part II** presents the extension to rigid fluid–structure interaction and morphing boundaries (chapter 4, **C2**) and the development of low-dimensional structural models for morphing aeroelastic applications (chapter 5, **C4**).
- **Part III** finally shows the three-dimensional aeroelastic application of conservative mesh adaptation with moving bodies and morphing boundaries to the analysis of nonclassical aileron buzz (chapter 6, **C1,C2**).

Finally, conclusions drawn from all parts are given jointly in chapter 7.

Part I

Numerical modeling of compressible flows with conservative mesh adaptation

2 Finite-volume ALE formulation of compressible fluid flows with conservative mesh adaptation

This chapter describes the `Flowmesh` solver for unsteady inviscid compressible flows developed in [75, 94, 140], which has been optimized, extended and employed throughout this work. Conservative mesh adaptation is performed through the link with the `MMG` remeshing library [45, 42]. Section 2.1 recalls the governing equations, section 2.2 introduces the edge-based finite-volume discretization, while sections 2.3 and 2.4 present the time integration algorithm. Mesh deformation is introduced in section 2.5, while the peculiar continuous time interpretation of local remeshing operations allowing for the conservative solution transfer between topology-changing adapted meshes is presented in section 2.6. Mesh adaptation strategies are recalled in section 2.7. Finally, novel contributions to the software implementation are briefly presented with a basic overview of data structure issues (section 2.8) and the optimization of search operations through hash functions (section 2.9).

2.1 Arbitrary Lagrangian–Eulerian formulation

As introduced in chapter 1, we consider the conservative Arbitrary Lagrangian–Eulerian formulation of inviscid compressible flows on a moving domain $\Omega(t)$ [46]

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)} \mathbf{u}\,\mathrm{d}\Omega + \oint_{\partial\Omega(t)} \hat{\mathbf{n}}\cdot(\mathbb{F}(\mathbf{u}) - \mathbf{v}\mathbf{u})\,\mathrm{d}\Gamma = \mathbf{0} \tag{2.1}$$

where $\mathbf{u}$ is the array of the conservative solution, composed by the mass density ρ, the momentum density $\rho\mathbf{U}$ and the total energy density ρe^t, and $\mathbb{F}(\mathbf{u})$ is its flux

$$\mathbf{u} = \begin{bmatrix} \rho \\ \rho\mathbf{U} \\ \rho e^t \end{bmatrix}, \qquad \mathbb{F}(\mathbf{u}) = \begin{bmatrix} \rho\mathbf{U} \\ \rho\mathbf{U}\otimes\mathbf{U} + P(\rho,e)\mathbb{I} \\ \rho e^t\mathbf{U} + P(\rho,e)\mathbf{U} \end{bmatrix} \tag{2.2}$$

Pressure $P(\rho,e)$ is computed through a suitable equation of state. The moving domain velocity is represented by the vector field $\mathbf{v}$.

Equations 2.1 are valid on each moving element $\Omega_k(t)$ in a triangulation of the domain $\Omega(t)$. Considering domain motion means both allowing the boundary to

move, i.e. $\partial\Omega(t)$ as in the case of aeroelastic simulations, but also allowing the movement of mesh elements $\Omega_k(t)$ with fixed domain boundaries, as it is often the case in mesh adaptation.

This equations are required to fulfill the Geometric Conservation Law

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)} \mathrm{d}\Omega = \oint_{\partial\Omega(t)} \hat{\mathbf{n}}\cdot\mathbf{v}\,\mathrm{d}\Gamma \tag{2.3}$$

which is considered as an additional constraint necessary for the consistency of the numerical results.

2.2 Node-pair finite-volume discretization

We consider a node-centered finite-volume discretization of the equations in the domain $\Omega(t)$. This is done by considering a simplicial mesh of the domain $\Omega(t)$, and defining the dual tessellation $\mathscr{T}_h(t)$ made by control cells $\mathscr{C}_i(t)$ surrounding each mesh node i (see for example figure 2.1). This leads to the definition of the control cell volumes $V_i(t)$ and the averaged solution $\mathbf{u}_i(t)$ as

$$V_i(t) \triangleq \int_{\mathscr{C}_i(t)} \mathrm{d}\Omega, \qquad \mathbf{u}_i(t) \triangleq \int_{\mathscr{C}_i(t)} \mathbf{u}(\mathbf{x},t)\,\mathrm{d}\Omega \tag{2.4}$$

Interfaces between control cells i and k, necessary for flux computations, are defined as

$$\partial\mathscr{C}_{ik}(t) \triangleq \partial\mathscr{C}_i(t) \cap \partial\mathscr{C}_k(t) \tag{2.5}$$

while control cells interfaces lying on the domain boundary are denoted as

$$\partial\mathscr{C}_i^{\partial}(t) \triangleq \partial\mathscr{C}_i(t) \cap \partial\Omega(t) \tag{2.6}$$

Following [145], this allows us to define the integrated normal vectors $\boldsymbol{\eta}_{ik}(t), \boldsymbol{\xi}_i(t)$ and the integrated normal velocities $\nu_{ik}(t), \nu_i^{\partial}(t)$ as

$$\begin{aligned} \boldsymbol{\eta}_{ik}(t) &\triangleq \int_{\partial\mathscr{C}_{ik}(t)} \hat{\mathbf{n}}_i\,\mathrm{d}\Gamma, \quad \nu_{ik}(t) \triangleq \int_{\partial\mathscr{C}_{ik}(t)} \hat{\mathbf{n}}_i\cdot\mathbf{v}\,\mathrm{d}\Gamma \\ \boldsymbol{\xi}_i(t) &\triangleq \int_{\partial\mathscr{C}_i^{\partial}(t)} \hat{\mathbf{n}}_i\,\mathrm{d}\Gamma, \quad \nu_i^{\partial}(t) \triangleq \int_{\partial\mathscr{C}_i^{\partial}(t)} \hat{\mathbf{n}}_i\cdot\mathbf{v}\,\mathrm{d}\Gamma \end{aligned} \tag{2.7}$$

This *node-pair* formulation [145] can encompass both finite-volume and finite element discretizations. In general, a node-pair is a couple of mesh nodes interacting in the numerical scheme. For finite-volume discretization on simplicial meshes, node-pairs simply correspond to mesh edges, while on other meshes it is possible to have node-pairs not overlapping with mesh edges[1]

[1] In two dimensions, an example is provided by a linear finite element discretization of a quadrangular element, making two opposite nodes interact in the numerical scheme (thus forming a node-pair) while not being connecting by a mesh edge [64].

It can be shown that, for a finite-volume to be closed and for the scheme to be conservative, the following relations hold [146]

$$\begin{aligned} \boldsymbol{\eta}_{ik}(t) &= -\boldsymbol{\eta}_{ki}(t) \\ \nu_{ik} &= -\nu_{ki} \\ \sum_{k \in \mathcal{K}_{i,\neq}} \boldsymbol{\eta}_{ik}(t) + \boldsymbol{\xi}_i(t) &= \mathbf{0} \end{aligned} \tag{2.8}$$

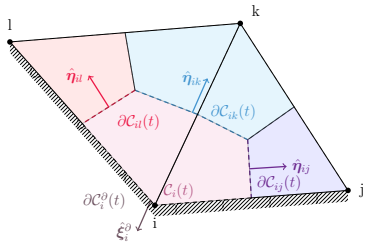


Figure 2.1 – Example of two-dimensional node-pair discretization.

where

$$\mathcal{K}_{i,\neq} = \{k : \partial\mathscr{C}_i \cap \partial\mathscr{C}_k \neq \emptyset\} \tag{2.9}$$

denotes the set of nodes adjacent to node i and distinct from it, so that looping on the this set means looping on the set of node-pairs connected to node i. For convenience, the integrated normal vectors norms η_{ik}, ξ_i and unit vectors $\hat{\boldsymbol{\eta}}_{ik}, \hat{\boldsymbol{\xi}}_i$ are straightforwardly defined as

$$\begin{aligned} \eta_{ik} &= ||\boldsymbol{\eta}_{ik}||, & \hat{\boldsymbol{\eta}}_{ik} &= \frac{\boldsymbol{\eta}_{ik}}{\eta_{ik}} \\ \xi_i &= ||\boldsymbol{\xi}_i||, & \hat{\boldsymbol{\xi}}_i &= \frac{\boldsymbol{\xi}_i}{\xi_i} \end{aligned} \tag{2.10}$$

Introducing a numerical approximation for fluxes (whose definition will be detailed in the following) as

$$\begin{aligned} \boldsymbol{\phi}_{ik}(\mathbf{u}_i, \mathbf{u}_k, \nu_{ik}, \boldsymbol{\eta}_{ik}) &\simeq \int_{\partial\mathscr{C}_{ik}} \hat{\mathbf{n}} \cdot (\mathbb{F}(\mathbf{u}) - \mathbf{v}\mathbf{u})\, \mathrm{d}\Gamma \\ \boldsymbol{\phi}_i^\partial(\mathbf{u}_i, \nu_i^\partial, \boldsymbol{\xi}_i) &\simeq \int_{\partial\mathscr{C}_i^\partial} \hat{\mathbf{n}} \cdot (\mathbb{F}(\mathbf{u}) - \mathbf{v}\mathbf{u})\, \mathrm{d}\Gamma \end{aligned} \tag{2.11}$$

allows us to write the node-pair finite-volume approximation of the governing equations in ALE form for each control cell $\mathscr{C}_i$ as

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}(V_i(t)\mathbf{u}_i) = -\sum_{k \in \mathcal{K}_{i,\neq}} \boldsymbol{\phi}_{ik}(\mathbf{u}_i, \mathbf{u}_k, \nu_{ik}, \boldsymbol{\eta}_{ik}(t)) - \boldsymbol{\phi}_i^\partial(\mathbf{u}_i, \nu_i^\partial, \boldsymbol{\xi}_i(t)) \\ \forall \mathscr{C}_i(t) \in \mathcal{T}_h(t) \end{aligned} \tag{2.12}$$

It is worth noticing that quantities $V_i(t)$, $\boldsymbol{\eta}_{ik}(t)$, $\boldsymbol{\xi}_i(t)$ are directly computable from the mesh at time instant t (typically an adapted mesh), so they will be assumed to be known at each time step and they will be used for the enforcement of the Geometric Conservation Law (GCL).

2.2.1 Swept volumes and Interface Velocity Consistency

As anticipated in chapter 1, the node-pair discretization of the GCL (eq. 2.3) will be used to compute the integrated interface normal velocities needed for the computation of the ALE numerical fluxes.

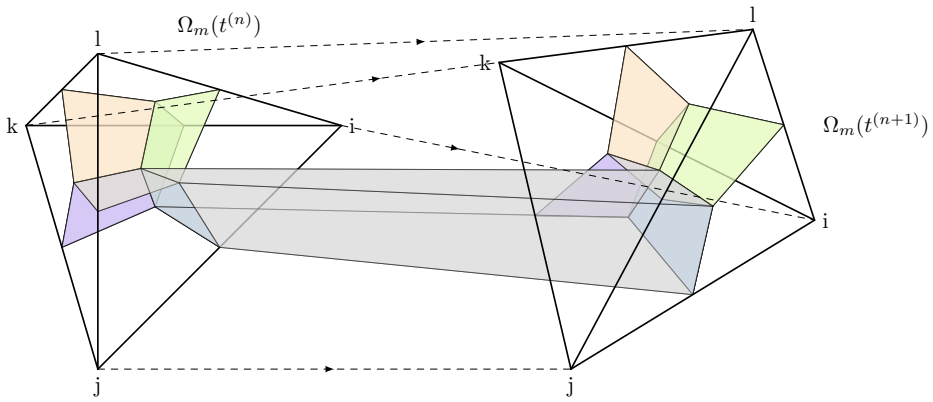


Figure 2.2 – Volume swept (grey) by the contribution brought by the tetrahedral element Ω_m to the interface Γ_{ij} (blue) during mesh modification with constant connectivity occurring in the interval $[t^{(n)}, t^{(n+1)}]$.

Defining the volume change $\Delta V_i(t;\tau)$ of cell i as a function of time, with respect to the past volume at time τ, in terms of volumes swept by its interfaces as

$$\Delta V_i(t;\tau) = V_i(t) - V_i(\tau) = \sum_{k \in \mathcal{K}_{i,\neq}} \Delta V_{ik}(t;\tau) + \Delta V_i^{\partial}(t;\tau) \tag{2.13}$$

allows to split the cell volume in a reference part, plus the volume change

$$V_i(t) = \sum_{k \in \mathcal{K}_{i,\neq}} \Delta V_{ik}(t;\tau) + V_i(\tau) \tag{2.14}$$

A three-dimensional example representation of the volume swept by a moving interface is given in figure 2.2, where the contribution brought by the portion of the interface $\Gamma_{ij}(t)$ in the element $\Omega_m(t)$ is considered. The above splitting of the cell volume is readily used to express the cell volume time derivative as

$$\frac{\mathrm{d}}{\mathrm{d}t} V_i(t) = \sum_{k \in \mathcal{K}_{i,\neq}} \frac{\mathrm{d}}{\mathrm{d}t} \Delta V_{ik}(t;\tau) + \frac{\mathrm{d}}{\mathrm{d}t} \Delta V_i^{\partial}(t;\tau) \tag{2.15}$$

It follows from calculus[2] that

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t} \Delta V_{ik}(t) &= \int_{\partial \mathscr{C}_{ik}(t)} \hat{\mathbf{n}}_i \cdot \mathbf{v} \, \mathrm{d}\Gamma \\ \frac{\mathrm{d}}{\mathrm{d}t} \Delta V_i^{\partial}(t) &= \int_{\partial \mathscr{C}_i^{\partial}(t)} \hat{\mathbf{n}}_i \cdot \mathbf{v} \, \mathrm{d}\Gamma \end{aligned} \tag{2.17}$$

[2]The volume ΔV swept by an arbitrary moving surface Γ can be computed through integration of the volume element swept by the area element $\mathrm{d}\Gamma$ in the infinitesimal time $\mathrm{d}t$, which reads [137]

$$\mathrm{d}V = \hat{\mathbf{n}} \cdot \mathbf{v} \, \mathrm{d}\Gamma \, \mathrm{d}t \tag{2.16}$$

where $\hat{\mathbf{n}}$ is the normal unit vector of the surface and $\mathbf{v}$ is the surface velocity.

When the node-pair discretization is introduced for the above relations, the following Interface Velocity Consistency (IVC) relations [126] are derived

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\Delta V_{ik}(t) &= \nu_{ik}(t) \\ \frac{\mathrm{d}}{\mathrm{d}t}\Delta V_i^{\partial}(t) &= \nu_i^{\partial}(t) \end{aligned} \tag{2.18}$$

The above relations, when replaced into equation 2.15, provide the following Discrete Geometric Conservation Law (DGCL) for each control cell

$$\frac{\mathrm{d}}{\mathrm{d}t}V_i(t) = \sum_{k\in\mathcal{K}_{i,\neq}} \nu_{ik}(t) + \nu_i^{\partial}(t), \qquad \forall \mathscr{C}_i(t) \in \mathscr{T}_h(t) \tag{2.19}$$

which is the node-pair discretization of the GCL (equation 2.3).

Finally, combining equation 2.12 with the IVC relations 2.18 (which provide a sufficient condition for the fulfillment of the DCGL 2.19), the semi-discrete system of equation is obtained

$$\begin{aligned} &\frac{\mathrm{d}}{\mathrm{d}t}\left(V_i(t)\mathbf{u}_i\right) = -\sum_{k\in\mathcal{K}_{i,\neq}} \boldsymbol{\phi}_{ik}(\mathbf{u}_i,\mathbf{u}_k,\nu_{ik},\boldsymbol{\eta}_{ik}(t)) - \boldsymbol{\phi}^{\partial}(\mathbf{u}_i,\nu_i^{\partial},\boldsymbol{\xi}_i(t)) \\ &\frac{\mathrm{d}}{\mathrm{d}t}\Delta V_{ik}(t) = \nu_{ik} \\ &\frac{\mathrm{d}}{\mathrm{d}t}\Delta V_i^{\partial}(t) = \nu_i^{\partial} \\ &\qquad\qquad\qquad\qquad \forall \mathscr{C}_i(t) \in \mathscr{T}_h(t),\ k \in \mathcal{K}_{i,\neq} \end{aligned} \tag{2.20}$$

Quantities $\Delta V_{ik}(t)$ and $\Delta V_i^{\partial}(t)$ are known from the mesh, together with quantities $V_i(t)$, $\boldsymbol{\eta}_{ik}(t)$, $\boldsymbol{\xi}_i(t)$, so the above system is complete in the unknowns $\mathbf{u}_i, \nu_{ik}, \nu_i^{\partial}$.

2.2.2 Metrics computation

After a dual cell is defined for each mesh node from contribution brought by each element connected to the given node, computation of integrated normal vectors follows from geometric considerations. Details about the two-dimensional computations can be found in the PhD thesis of D. Isola [93], while extension to the three-dimensional case can be found in the PhD thesis of B. Re [141].

2.2.3 Domain numerical fluxes

In this work, a numerical approximation for solution flux across domain interfaces

$$\boldsymbol{\phi}_{ik}(\mathbf{u}_i,\mathbf{u}_k,\nu_{ik},\boldsymbol{\eta}_{ik}) \simeq \int_{\partial\mathscr{C}_{ik}(t)} \hat{\mathbf{n}}\cdot(\mathbb{F}(\mathbf{u}) - \mathbf{v}\mathbf{u})\,\mathrm{d}\Gamma \tag{2.21}$$

is defined using a Total Variation Diminishing (TVD) approach [162]. Thus, a second-order centered approximation $\mathbf{\Phi}^{II}$ and a first-order Roe approximate Riemann solver $\mathbf{\Phi}^{I}$ [109, 108, 160])

$$\begin{aligned}\mathbf{\Phi}^{II}(\mathbf{u}_i,\mathbf{u}_k,\boldsymbol{\eta}_{ik},\nu_{ik}) &\triangleq \boldsymbol{\eta}_{ik}\cdot\frac{\mathbb{F}(\mathbf{u}_i)+\mathbb{F}(\mathbf{u}_k)}{2}-\nu_{ik}\cdot\frac{\mathbf{u}_i+\mathbf{u}_k}{2}\\ \mathbf{\Phi}^{I}(\mathbf{u}_i,\mathbf{u}_k,\boldsymbol{\eta}_{ik},\nu_{ik}) &\triangleq \boldsymbol{\eta}_{ik}\cdot\frac{\mathbb{F}(\mathbf{u}_i)+\mathbb{F}(\mathbf{u}_k)}{2}-\nu_{ik}\cdot\frac{\mathbf{u}_i+\mathbf{u}_k}{2}-\frac{1}{2}|\tilde{\mathbf{A}}|\,(\mathbf{u}_k-\mathbf{u}_i)\end{aligned} \tag{2.22}$$

are blended with a flux-limiter approach so that the first order, monotonicity preserving scheme is employed near solution discontinuities, while the second order scheme is used in smooth flow regions.

The Roe matrix $\tilde{\mathbf{A}}$ is built using the Jacobian of the flux function projected along the normal direction $\hat{\boldsymbol{\eta}}_{ik}$ evaluated at the intermediate Roe state $\tilde{\mathbf{u}}(\mathbf{u}_i,\mathbf{u}_k)$. Intermediate state $(\tilde{\rho},\tilde{\mathbf{m}},\tilde{h}^t)$ for a polytropic ideal gas reads

$$\begin{aligned}\tilde{\mathbf{m}} &= \frac{\mathbf{m}_i\sqrt{\rho_k}+\mathbf{m}_k\sqrt{\rho_i}}{\sqrt{\rho_i}+\sqrt{\rho_k}}\\ \tilde{h}^t &= \frac{h_i^t\sqrt{\rho_k}+h_k^t\sqrt{\rho_i}}{\sqrt{\rho_i}+\sqrt{\rho_k}}\end{aligned} \tag{2.23}$$

and $\tilde{\rho}=\sqrt{\rho_i\rho_k}$ as the Roe matrix is independent from its value for an ideal polytropic gas (an extension to van der Waals gas can be found in [76]).

Matrix $|\tilde{\mathbf{A}}|$ is built taking the eigenvalue decomposition of matrix $\tilde{\mathbf{A}}$ and replacing signed eigenvalues by their absolute values

$$\tilde{\mathbf{A}}=\tilde{\mathbf{R}}\tilde{\mathbf{\Lambda}}\tilde{\mathbf{L}} \tag{2.24}$$

$$|\tilde{\mathbf{A}}|=\tilde{\mathbf{R}}|\tilde{\mathbf{\Lambda}}|\tilde{\mathbf{L}} \tag{2.25}$$

where $|\Lambda(\tilde{\mathbf{u}},\nu_{ik},\eta_{ik}(t))|$ is the diagonal matrix built with the absolute values of the integrated eigenvalues $\tilde{\lambda}=\lambda(\tilde{\mathbf{u}},\nu_{ik},\eta_{ik}(t))$ in three dimensions

$$\lambda(\tilde{\mathbf{u}},\nu_{ik},\boldsymbol{\eta}_{ik}(t))=\begin{bmatrix}\boldsymbol{\eta}_{ik}(t)\cdot\frac{\tilde{\mathbf{m}}}{\tilde{\rho}}-\nu_{ik}+c(\tilde{\mathbf{u}})\eta_{ik}(t)\\ \boldsymbol{\eta}_{ik}(t)\cdot\frac{\tilde{\mathbf{m}}}{\tilde{\rho}}-\nu_{ik}\\ \boldsymbol{\eta}_{ik}(t)\cdot\frac{\tilde{\mathbf{m}}}{\tilde{\rho}}-\nu_{ik}\\ \boldsymbol{\eta}_{ik}(t)\cdot\frac{\tilde{\mathbf{m}}}{\tilde{\rho}}-\nu_{ik}\\ \boldsymbol{\eta}_{ik}(t)\cdot\frac{\tilde{\mathbf{m}}}{\tilde{\rho}}-\nu_{ik}-c(\tilde{\mathbf{u}})\eta_{ik}(t)\end{bmatrix} \tag{2.26}$$

When the eigenvalues are close to zero, the numerical dissipation provided by the upwind term in the Roe flux can be too small, and the Roe linearization may fail to satisfy the entropy condition, necessary for the convergence to a unique physically relevant weak solution [109]. For this reason, an entropy fix is employed to provide

a suitable modification of the eigenvalues to avoid the problem [131]. In this work, the entropy fix proposed in [145] is employed

$$\hat{\lambda}_p = \begin{cases} |\tilde{\lambda}_p| & \text{if } |\tilde{\lambda}_p| > \tilde{\delta} \\ \dfrac{\tilde{\lambda}_p^2 + \tilde{\delta}^2}{2\tilde{\delta} + \epsilon} & \text{if } |\tilde{\lambda}_p| < \tilde{\delta} \end{cases} \tag{2.27}$$

where $\epsilon = 10^{-12}$ is a small parameter to avoid division by zero, and $\tilde{\delta}$ is a function of the Mach number M evaluated at the intermediate state as

$$\begin{aligned} M(\tilde{\mathbf{u}}, \nu_{ik}, \boldsymbol{\eta}_{ik}(t)) &= \frac{1}{c(\tilde{\mathbf{u}})}\left(\frac{\boldsymbol{\eta}_{ik}(t)\cdot\tilde{\mathbf{m}}}{\tilde{\rho}} - \frac{\nu_{ik}}{\eta_{ik}(t)}\right) \\ \tilde{\delta} &= \frac{1}{5}\left(M(\tilde{\mathbf{u}}, \nu_{ik}, \eta_{ik}(t)) + 1\right) \end{aligned} \tag{2.28}$$

Flux limiter. A flux limiter $\boldsymbol{\Gamma}$ is used to blend first order and second order scheme in the definition of the numerical flux

$$\phi_{ik} = \phi_{ik}^I + \boldsymbol{\Gamma}\left(\phi_{ik}^{II} - \phi_{ik}^I\right) = \phi_{ik}^{II} + \frac{1}{2}\tilde{\mathbf{R}}|\Lambda|\left(\boldsymbol{\Gamma} - \mathbf{I}^5\right)\tilde{\mathbf{L}}(\mathbf{u}_k - \mathbf{u}_i) \tag{2.29}$$

The diagonal flux limiter $\boldsymbol{\Gamma}$ is defined according to the definition given by van Leer [162]. Limiters typically use the ratio of consecutive gradients over an unstructured mesh in the attempt to measure the smoothness of flow. In the node-pair framework, this is achieved through the extended node-pair structure [77]. Taking into consideration the node-pair $i-k$, their extension $i^*-i-k-k^*$ is build by finding the edges i^*-i and $k-k^*$ which are best aligned in the direction $i-k$. This allows the definition of the p-component of the characteristic jump as

$$\tilde{\mathbf{q}}_p = \begin{cases} \dfrac{\boldsymbol{\eta}_{ik}\cdot(\mathbf{x}_k - \mathbf{x}_i)}{\boldsymbol{\eta}_{ik}\cdot(\mathbf{x}_{k^*} - \mathbf{x}_k)}\tilde{\mathbf{L}}_p(\mathbf{u}_{k^*} - \mathbf{u}_k) & \text{if } \tilde{\lambda}_p > 0 \\ \dfrac{\boldsymbol{\eta}_{ik}\cdot(\mathbf{x}_k - \mathbf{x}_i)}{\boldsymbol{\eta}_{ik}\cdot(\mathbf{x}_i - \mathbf{x}_{i^*})}\tilde{\mathbf{L}}_p(\mathbf{u}_i - \mathbf{x}_{i^*}) & \text{if } \tilde{\lambda}_p \leq 0 \end{cases} \tag{2.30}$$

where $\tilde{\mathbf{L}}_p$ is the $p-$th row of the left eigenvector matrix.

2.2.4 Boundary numerical fluxes

Boundary conditions are imposed in weak form, so boundary numerical fluxes are defined as the evaluation of the numerical flux $\boldsymbol{\phi}_i^\partial$ in a specified boundary state $\mathbf{u}_i^\partial(\mathbf{u}_i, \boldsymbol{\xi}_i^\partial, \nu_i^\partial)$.

Slip (wall) boundary conditions are imposed through the definition of a boundary state characterized by the relative wall-normal flow velocity $(\mathbf{m}_i \cdot \hat{\boldsymbol{\xi}}_i)/\rho_i - \mathbf{v}\cdot\hat{\boldsymbol{\xi}}_i$, so

$$\mathbf{u}_i^{\partial,W} = \mathbf{u}_i - \begin{bmatrix} 0 \\ \left(\mathbf{m}_i\cdot\hat{\boldsymbol{\xi}}_i - \frac{\phi_i\nu_i}{\xi_i}\right)\hat{\boldsymbol{\xi}}_i \\ \frac{1}{2}\rho_i\left|\mathbf{m}_i\cdot\hat{\boldsymbol{\xi}}_i - \frac{\phi_i\nu_i}{\xi_i}\right|^2 \end{bmatrix} \tag{2.31}$$

and its numerical flux reads

$$\mathbf{\Phi}_i^{\partial,W} = \Pi(\mathbf{u}_i) \begin{bmatrix} 0 \\ \boldsymbol{\xi}_i \\ \nu_i^{\partial} \end{bmatrix} \tag{2.32}$$

Non-reflecting boundary conditions are based on a characteristic directions analysis to impose inflow or outflow conditions on each characteristic field [77]. This leads to the following definition of the boundary state

$$\mathbf{u}_i^{\partial,\infty}(\mathbf{u}_i, \nu_i^{\partial}, \boldsymbol{\xi}_i) = \mathbf{u}_i + \mathbf{R}(\mathbf{u}_i, \boldsymbol{\xi}_i)\mathscr{SN}(\mathbf{u}_i, \nu_i^{\partial}, \boldsymbol{\xi}_i)\mathbf{L}(\mathbf{u}_i, \boldsymbol{\xi}_i)(\mathbf{u}_\infty - \mathbf{u}_i) \tag{2.33}$$

where $\mathscr{SN}$ selects only the parts of matrix $\mathbf{\Lambda}$ with strictly negative eigenvalues, and $\mathbf{u}_\infty$ is the prescribed asymptotic flow state.

2.3 BDF time integration

The semi-discrete flow equations 2.20 are discretized in time through Backward Differentiation Formulæ [139]. Thus, the approximation of the time derivative of a function y or order $p+1$ reads

$$\frac{\mathrm{d}y}{\mathrm{d}t} \simeq \frac{1}{\Delta t} \sum_{q=-1}^{p} a_q y^{n-q} \tag{2.34}$$

Since the DGCL relation (equation 2.18) involves swept volumes between different time instants, it is convenient to recast this formula by defining

$$\Delta y^n = y^n - y^{n-1}, \qquad \alpha_q = \sum_{d=-1}^{q} a_d \tag{2.35}$$

leading us to the equivalent formula

$$\frac{\mathrm{d}y}{\mathrm{d}t} \simeq \frac{1}{\Delta t} \sum_{q=-1}^{p-1} \alpha_q \Delta y^{n-q} \tag{2.36}$$

Using relations 2.34, 2.36 for the approximation of the semi-discrete equations 2.20 leads to the fully discrete system

$$\begin{aligned}
\sum_{q=-1}^{p} a_q V_i^{n-q} \mathbf{u}_i^{n-q} &= -\Delta t \left[\sum_{k \in \mathscr{K}_{i,\neq}} \boldsymbol{\phi}_{ik}(\mathbf{u}_i, \mathbf{u}_k, \nu_{ik}, \boldsymbol{\eta}_{ik}(t)) + \boldsymbol{\phi}_i^{\partial}(\mathbf{u}_i, \nu_i^{\partial}, \boldsymbol{\xi}_i(t)) \right]^{n+1} \\
\sum_{q=-1}^{p} \alpha_q \Delta V_{ik}^{n-q} &= \Delta t \nu_{ik}^{n+1} \\
\sum_{q=-1}^{p} \alpha_q \Delta V_i^{\partial\, n-q} &= \Delta t \nu_i^{\partial\, n+1}
\end{aligned} \tag{2.37}$$

where $[\boldsymbol{\phi}(\cdot)]^{n+1}$ denotes the implicit evaluation of numerical fluxes at time $t^{(n+1)}$. The above nonlinear system can in principle be solved by a Newton-Raphson procedure, but the poor diagonal dominance poses problems to the convergence of the procedure. In order to improve the diagonal dominance of the equations, an iterative solution procedure is devised by resorting to a dual time stepping procedure [164] combined with an inexact Newton-Raphson method (the defect-correction method [99]).

The same system of equations 2.37 is applied to the case of dynamically adaptive meshes, i.e. meshes with variable topology. The passage from constant and variable topology leads to a broader definition of the node-pairs set $\mathcal{K}_{i,\neq}$, which is allowed to vary in time by means of a dynamic data structure, as it will be shown in section 2.6.

2.4 Dual time stepping

System of equations 2.37 can be recast in matrix notation as

$$\mathbf{R}^*(\mathbf{U}^{n+1}) = \frac{a_{-1}}{\Delta t}\mathbf{V}^{n+1}\mathbf{U}^{n+1} + \mathbf{S} + \mathbf{R}(\mathbf{U}^{n+1}) = \mathbf{0} \tag{2.38}$$

where $\mathbf{U} = (\mathbf{u}_1, \dots \mathbf{u}_{N_v})^T$ is the average cell solution vector, $\mathbf{V} = (V_1, \dots V_{N_v})^T$ is the control volumes vector, $\mathbf{R}$ is the vector containing all numerical flux contributions at time $t^{(n+1)}$ and $\mathbf{S} = \frac{1}{\Delta t}\sum_{q=0}^{p} a_q \mathbf{V}^{n-q}\mathbf{U}^{n-q}$ is treated as a source term. Nonlinearity of the above system can in principle be tackled through a Newton-Raphson procedure, but the poor diagonal dominance of the system makes the convergence difficult [34].

Dual time stepping aims at solving the nonlinear system by introducing a time derivative in the fictitious time τ

$$\frac{\mathrm{d}\mathbf{U}^{n+1}}{\mathrm{d}\tau} + \mathbf{R}^*(\mathbf{U}^{n+1}) = \mathbf{0} \tag{2.39}$$

which is readily approximated through a backward difference formula and combined with a Newton approximation of the implicit term, so that a sequence of approximants $\mathbf{U}^m$ for $\mathbf{U}^{n+1}$, starting from $\mathbf{U}^0 = \mathbf{U}^n$, is obtained

$$\left(\frac{1}{\Delta\tau} + \frac{\partial \mathbf{R}^*}{\partial \mathbf{U}^T}\right)(\mathbf{U}^{m+1} - \mathbf{U}^m) + \mathbf{R}^*(\mathbf{U}^{n+1}) = \mathbf{0} \tag{2.40}$$

After expanding the expression of $\mathbf{R}^*$, the iteration formula is obtained

$$\left(\frac{1}{\Delta\tau} + \frac{a_{-1}}{\Delta t}\mathbf{V}^{n+1} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}^T}\right)(\mathbf{U}^{m+1} - \mathbf{U}^m) = -\frac{a_{-1}}{\Delta t}\mathbf{V}^{n+1}\mathbf{U}^m - \mathbf{S} - \mathbf{R}(\mathbf{U}^m) = \mathbf{0} \tag{2.41}$$

The exact numerical flux Jacobian $\partial \mathbf{R}/\partial \mathbf{U}^T$ is replaced with an approximate first-order one (making the method a quasi-Newton scheme [134]), in order to increase the diagonal dominance. Details of the computation can be found in [34, 93]. Finally, the linear system is solved by means of a symmetric Gauss-Seidel method [34].

Local time stepping. In order to satisfy the Courant-Friedrichs-Lewy (CFL) stability condition for linear hyperbolic equations, stating that the Courant number Co should not be greater than 1, a local time stepping technique is used to define a local pseudo-time step $\Delta\tau_i$ such that the CFL condition is locally satisfied at each time step

$$\Delta\tau_i = \frac{V_i \mathrm{Co}}{\sum_{k\in\mathcal{K}_{i,\neq}} \lambda_{\max}(\mathbf{u}_i, \mathbf{u}_k, \boldsymbol{\eta}_{ik}, \nu_{ik}) + \lambda_{\max}(\mathbf{u}_i, \boldsymbol{\xi}_i, \nu_i^{\partial})} \tag{2.42}$$

where $\lambda_{\max}$ is the eigenvalue of the Roe matrix with largest absolute value. To increase the convergence speed, the Courant number is increased when the residual norm is decreasing through the update formula

$$\mathrm{Co}^m = \min\left(\max\left(\gamma \frac{||\mathbf{R}(\mathbf{U}^{m-1})||_{L^2}}{||\mathbf{R}(\mathbf{U}^m)||_{L^2}}, 1\right)\mathrm{Co}^{m-1}, \mathrm{Co}^{\max}\right) \tag{2.43}$$

where γ is a user defined increase ratio, and $\mathrm{Co}_{\max}$ is an upper bound on the Courant number.

2.5 Mesh deformation

Before performing mesh adaptation at time $t^{(n+1)}$, the position of moving boundaries need to be updated according to a prescribed time law or to the structural dynamics of the body they are attached to (the cases related to this work will be showed in chapter 4). In order to preserve a body-fitted mesh at each time instant, mesh nodes at time $t^{(n+1)}$ need to be relocated so that they comply with this new boundary position.

A standard approach proposed in [17] to preserve a body-fitted mesh with boundary motion is to employ a structural analogy and to consider the fluid flow domain as a fictitious elastic solid, discretized on the same mesh, whose unknown displacement is treated in a Lagrangian way. A schematic representation of the typical time advancement of the fluid flow solver with moving boundaries and dynamic mesh adaptation is shown in algorithm 1, where mesh modifications occurring at each time step are shown in italic (fluid-structure interaction is not considered yet).

Algorithm 1 Schematic time loop for unsteady boundaries and mesh adaptation.

```
Read u^(0);                                          /* Initial condition */
Read T;                                              /* Set maximum time */
Read N_max;                                          /* Set nb. of time steps */
Set Δt = T/N;                                        /* Set time step */
for n = 0, ..., N − 1 do                             /* Time loop */
    Update boundary position at t^(n+1);
    Deform mesh;
    Predict solution on unadapted mesh;
    for m = 1, ..., M_adapt do                       /* (Multiple) Mesh adaptation */
        Conservative mesh adaptation;
    end for
    Fluid flow solution on adapted mesh;
end for
```

At each time step, a brand new elastic problem is considered, and the mesh $\Omega^{(n)} = \Omega\left(t^{(n)}\right)$ at time $t^{(n)}$ is considered as the reference ("undeformed") configuration. Defining the unknown displacements

$$\boldsymbol{\delta}^{(n+1)} \triangleq \mathbf{x}(t^{(n+1)}) - \mathbf{x}(t^{(n)}) \tag{2.44}$$

the variational formulation reads

$$\int_{\Omega^{(n)}} \boldsymbol{\epsilon}(\mathbf{v}) : \boldsymbol{\sigma}(\boldsymbol{\delta}^{(n+1)})\, \mathrm{d}\Omega = 0 \qquad \forall \mathbf{v} \in [H^1(\Omega^{(n)})]^d \tag{2.45}$$

with the Dirichlet boundary conditions

$$\begin{aligned} \boldsymbol{\delta}^{(n+1)} &= \mathbf{x}(t^{(n+1)}) - \mathbf{x}(t^{(n)}) && \text{on } \Gamma_D^{(n)} \\ \boldsymbol{\delta}^{(n+1)} &= \mathbf{0} && \text{on } \partial\Omega^{(n)} \backslash \Gamma_D^{(n)} \end{aligned} \tag{2.46}$$

With the infinitesimal deformation hypothesis, the small strain tensor ϵ reads

$$\boldsymbol{\epsilon}(\mathbf{v}) = \frac{1}{2}\left(\boldsymbol{\nabla}\mathbf{v} + (\boldsymbol{\nabla}\mathbf{v})^T\right) \tag{2.47}$$

while the Cauchy stress tensor $\boldsymbol{\sigma}$ for a linear elastic material reads

$$\boldsymbol{\sigma}(\mathbf{v}) = 2\mu\boldsymbol{\epsilon}(\mathbf{v}) + \lambda \mathrm{tr}(\boldsymbol{\epsilon}(\mathbf{v}))\mathbf{I} \tag{2.48}$$

where μ, λ are the Lamé coefficients, typically expressed as functions of the elastic modulus and the Poisson coefficient.

Variable stiffness. Since adapted meshes are typically non-uniform, with highly refined zones near solid bodies and specific flow patterns (like shock waves), various strategies have been studied to increase the stiffness of smaller elements so to avoid mesh tangling (the occurrence of invalid elements with null or negative volume). In this work, a variable stiffness approach is used, where a local elastic modulus E_k inversely proportional to the minimum edge length on each element Ω_k is assigned as

$$E_k = \frac{1}{\min_{i,j \in \mathcal{K}_k} ||\mathbf{x}_i - \mathbf{x}_j||^\beta} \tag{2.49}$$

where $\mathcal{K}_k$ is the set of nodes belonging to the element k, and β is a tuning coefficient (in this work, a value between 2.0 and 3.0 is used).

Intersections of moving and fixed boundaries in three dimensions. In three-dimensions, we can be faced with the intersection of a moving boundary (like a rigidly-moving aircraft wing) with a fixed one (like a wall at the wing root). A strategy to deal with this case is to consider a hierarchical application of the elastic analogy, according to the following steps.

1. Compute moving boundary displacement at time $t^{(n+1)}$.
2. Identify one-dimensional intersections of moving bodies with fixed boundaries, then apply the two-dimensional elastic analogy on the fixed boundary, given the displacement of the one-dimensional intersection.
3. Apply the elastic analogy in the three-dimensional domain, given the computed displacement on the two-dimensional boundary.

This methodology is currently implemented for planar fixed boundaries.

Time step split for large movements. When large boundary movements are required, it could happen that it is not possible to perform mesh deformation without producing invalid elements. In this case, it is useful to split the time step in several sub-steps and to solve several elastic problems successively, in order to ease the required boundary displacement. In order to illustrate the procedure between time $t^{(n)}$ and time $t^{(n+1)}$, we introduce a time cursors τ keeping track of the successfully elapsed time (starting from $t^{(n)}$), and a time cursor t representing the current tentative time (ending at $t^{(n+1)}$). A maximum of $i_{\max}$ total sub-steps is allowed, while a maximum of $j_{\max}$ consecutive time reductions is allowed. A pseudocode is shown in algorithm 2.

Parameters $\alpha^{(-)}, \alpha^{(+)}$ determine the geometric progression of the time step decrease or increase, respectively. In this work, the values $\alpha^{(-)} = 0.5$ and $\alpha^{(+)} = 1.1$

Algorithm 2 Time step split for large movements.

```
1: τ = t^(n)                                          /* Set old time cursor */
2: t = t^(n+1)                                        /* Set new time cursor */
3: Δτ = t − τ                                         /* Set time step */
4: j = 0                                              /* Initialize split counter */
5: for i = 0, ..., i_max − 1 do
6:     Update movement;
7:     Deform mesh;
8:     if Invalid elements then
9:         Reset movement;
10:        if j ≥ j_max then
11:            Return failure;
12:        else
13:            j+ = 1;
14:            t = min(τ + α^(−)Δτ, t^(n+1))          /* Decrease new time */
15:            Δτ = t − τ                             /* Set decreased time step */
16:        end if
17:    else
18:        τ = t                                      /* Update old time */
19:        if τ = t^(n+1) then
20:            Return success;
21:        else
22:            t = min(τ + α^(+)Δτ, t^(n+1))          /* Increase new time */
23:            Δτ = t − τ                             /* Set increased time step */
24:        end if
25:        j = 0;
26:    end if
27: end for
28: if i = i_max then
29:    Return failure;
30: end if
```

have been used. Functions *Update movement* and *Reset movement* compute the new boundary displacement or reset it to the previously valid values, respectively. Deform mesh solves the elastic problem. In case of failure of the algorithm (for exceeding the maximum value of sub-steps or successive time step reductions), the remeshing library is called to improve the quality of the starting mesh at time $t^{(n)}$, then the procedure is re-applied.

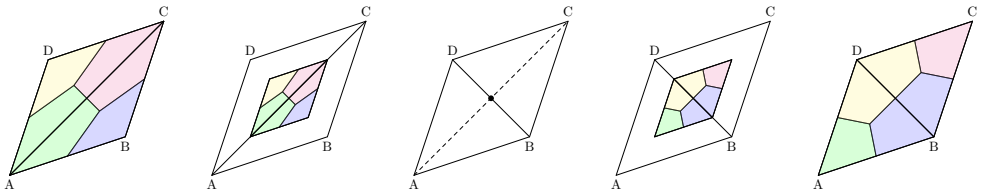
2.6 Conservative ALE scheme with variable topology

Continuous time interpretation of local remeshing operations. Dynamic mesh adaptation changes the topology of a mesh by introducing or removing nodes and edges, thus inserting or removing control cells and interfaces in the finite-volume discretization. As outlined in chapter 1, solution conservation is not automatically guaranteed unless specific numerical schemes are employed. In constant-topology meshes, the ALE formulation allows to enforce conservation through the computation geometrically-consistent interface velocities by means of the DGCL.

The methodology developed in [75, 94, 140] appears as an extension of the ALE approach to variable topology meshes. The very same Interface Velocity Consistency method used to compute interface velocities, while fulfilling the DGCL, can be applied to build a conservative solution transfer method which avoids solution interpolation from the old to the new mesh. In order to do that, it is necessary to introduce a continuous interpretation of each local mesh modification, so that the motion of each interface can be tracked in time as new control cells grow from null volumes, and removed control cells shrink to null volumes from time $t^{(n)}$ to time $t^{(n+1)}$. In this way, swept volumes can be computed and used for the computation of mesh interface velocities for every local mesh modification operation. Discrete solutions in null control cells persist as degrees of freedom as long as they are needed by a p-step time integration scheme, so that their value can contribute to the computation of numerical fluxes.

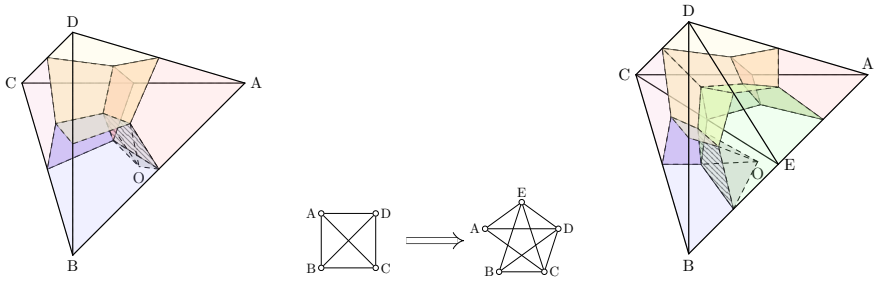
Three-steps continuous time procedure. For a correct computation of swept volumes with connectivity change, the continuous time interpretation of local mesh modifications needs to be applied to the set of elements sharing the edge interested by a modification (collapse, split...). A general procedure relies in splitting the local mesh modification occurring between time $t^{(n)}$ and time $t^{(n+1)}$ in three steps.

1. The elements sharing the edge collapse from their previous (unadapted) configuration to an arbitrary point; collapse swept volumes are computed.



a $\tau = 0$. b $\tau = 0.25$. c $\tau = 0.5$. d $\tau = 0.75$. e $\tau = 1$.

Figure 2.3 – Three-steps procedure for a two-dimensional edge swap.



a $t^{(n)}$. b Connectivity change. c $t^{(n+1)}$.

Figure 2.4 – Three-steps procedure for a three-dimensional edge split.

2. The connectivity is changed at fixed time (no swept volumes).

3. The elements are re-expanded to their current (adapted) configuration; expansion swept volumes are computed.

The choice of the collapse point is arbitrary, and computation of swept volumes both in the collapse and expansion phase effectively filters out all contributions unrelated to the mesh modification operation.

An example for a two-dimensional edge-swap operation is given in figure 2.3. For illustration purposes, a fictitious time τ going from 0 to 1 in the time range $[t^{(n)}, t^{(n+1)}]$ is considered. Figure 2.4 shows an example of a three-dimensional edge split operation (only one element sharing the split edge is shown).

Handling of variable topology. The dynamic nature of the adaptive mesh is handled by allowing for a time-varying definition of the set of adjacent nodes $\mathcal{K}_{i,\neq}^{(n+1)}$ collecting all cell interfaces present in the mesh at time $t^{(n+1)}$. Since also null cell volumes contribute to the numerical scheme through the volumes swept by their interfaces during their collapse (in case of node deletion) or expansion (in

case of node insertion), the additional set of node-pairs needs to be considered

$$\mathcal{K}_{i,\neq}^{[n-p,n+1)} = \{k \in \mathcal{K}, k \notin \mathcal{K}_{i,\neq}^{n+1} \quad \text{s.t.} \quad v_{ik}^{n+1} \neq 0\} \tag{2.50}$$

collecting moving interfaces between cells in the mesh at time $t^{(n+1)}$ and cells destroyed or created in the time range $[t^{(n-p)}, t^{(n+1)})$. Handling of this time-varying mesh connectivity needs the usage of specific dynamic data structures; an overview of the data structures employed in this work will be given in section 2.8.

Local remeshing operations The conservative procedure described above can in principle be applied to an arbitrary mesh modification procedure. In `Flowmesh`, mesh modification is performed through the link, introduced by B. Re in her PhD work [141], with the `Mmg` remeshing library developed by C. Dobrzynski and coworkers [45, 42]. The `Mmg` library [43] is a complete tool for simplicial remeshing, supporting both two-dimensional, three-dimensional and surface remeshing. Consistently with the metric-based mesh adaptation framework presented in chapter 1, mesh modification is driven by a user-defined scalar (isotropic) or tensor (anisotropic) *metric* field. Local mesh modification operations are performed if they lead to an increase in the mesh quality measure.

Beside the metric field, the user can provide a minimum and maximum edge size. Element size variation is mitigated by imposition of a graded edge size variation. Given two consecutive edges $\mathbf{e}_1, \mathbf{e}_2$, their length variation should satisfy the relation

$$\frac{1}{h_{\text{grad}}} \leq \frac{|\mathbf{e}_1|}{|\mathbf{e}_2|} \leq h_{\text{grad}} \tag{2.51}$$

where h_{grad} is a gradation control parameter. Mesh modification on a curved surface is performed after the surface geometry as been partitioned and its geometry represented by means of cubic Bézier curves. The accuracy of the geometrical approximation can be regulated by setting the maximum Hausdorff distance [42], providing a measure of the distance between the ideal geometry and the approximated one.

A sequence of local mesh modification operation is performed by the remeshing library each time it is called, i.e. at each time step. Being the main purpose of the remeshing library that of returning a valid mesh at the end of each call, no history of local remeshing operations (needed for swept volumes computation) is automatically provided through API functions. Typically, entities in the output mesh will not preserve any explicit mapping with entities in the input mesh, even for persisting entities not removed in the remeshing process, since they will be typically referenced by different indices in the data structures. A minimally invasive tracking of local mesh modification operation has been implemented in [141] by means of callback functions, allowing to update the mesh connectivity

in `Flowmesh` by inserting/deleting mesh entities while preserving references to persistent mesh entities, so that swept volumes can be computed and contribute to the correct finite-volume cell solution.

A brief summary of the local mesh modifications implemented in `Mmg` [42] and supported in `Flowmesh` is reported in the following.

Node deletion. This operation is implemented as an edge collapse. One end of the edge is collapsed onto the other one, and their connectivities are merged. As a result, all the elements once sharing the collapsed edge are also deleted.

Node insertion. A node can be inserted by means of three different operations: Edge split, element split, Delauney node insertion. Edge split inserts a new node in the barycenter of an edge; in this case, the original edge (and the node-pair between its ends) is deleted and two new edges (and node-pairs) are created between the new node and the old ones. Element split inserts a new element in the barycenter of an element; old edges (and node-pairs) of the tetrahedra are untouched by the operation, but new edges (and node-pairs) are created between the new node and the old tetrahedra vertices. The last operation uses Delauney triangulation to completely reconstruct the connectivity of a new point p, whose position has been identified according to metric field and quality measure criteria. Firstly, a cavity surrounding point p is defined by deleting the elements whose circumsphere contains the new point p. Then, new tetrahedra are created by connecting the new node to the boundary of the cavity.

Edge swap. A three-dimensional extension of the classical two-dimensional edge-swap operation can be performed by interpreting it as a sequence of edge split and edge collapse operations. Firstly, the edge to be swapped is split. Then, its two halves are collapsed, leaving the swapped configuration as a result.

Node relocation. A number of regularization techniques, such as barycentric regularization [65], aim at improving mesh spacing by moving nodes while keeping their connectivity unchanged. In this respect, these techniques are analogous to mesh deformation techniques, and the simple ALE formulation is sufficient to recover solution conservation.

Special care in node-pair reactivation Since a sequence of local mesh modification operations is performed at each time step, it is possible for the remeshing library to re-create an edge that was previously deleted during the same time step. This is the case, for example, of an edge split operation inserting a new node k between nodes i, j followed by the collapse of either of edges $i-k$ or $j-k$, recreating

a connection between nodes i and j. In this case, it is necessary to recognize that the new edge $i-j$ contributes to the same node-pair $i-j$ already existing in the discretization, so that its existence is preserved. This operation needs to search a node-pair by its nodes, and it can needs some programming care in order to be performed without significant performance penalties 2.9.

2.7 Mesh adaptation strategy

As introduced in chapter 1, a metric-based approach is used in this work for converting an error indicator field into an edge size map to be used for remeshing purposes. Construction of an isotropic size map is described in section 2.7.1, while an anisotropic size map is introduced in section 2.7.2.

2.7.1 Isotropic mesh adaptation

Following a feature-based approach, the error indicator employed in this work is built using first and second derivatives of a scalar component of the conservative solution or a related quantity (like Mach number, or entropy, or vorticity magnitude) in order to target the most relevant flow features, without any attempt of a mathematical error estimation. In this work, an error estimator e_i based on the gradient of a quantity p_i in the control cell Ω_i is defined as

$$e_i(p_i) = V_i^{1/d} ||\boldsymbol{\nabla} p_i|| \tag{2.52}$$

where multiplication by a power of the cell volume (with $d = 2,3$ as the space dimensions) contributes to smoothen sharp variations and gives the same physical dimensions to both p_i and e_i. A Hessian-based indicator is defined through the projection of the Hessian matrix $\mathbf{H}(p_i)$ along the tangent $\hat{\boldsymbol{\tau}}$ and normal $\hat{\boldsymbol{\xi}}$ directions to the fluid flow, as

$$e_i(p_i) = V_i^{2/d} \sqrt{E(\hat{\boldsymbol{\tau}}, p_i)^2 + E(\hat{\boldsymbol{\xi}}, p_i)^2}, \qquad E(\hat{\mathbf{n}}, p_i) = \hat{\mathbf{n}}^T \mathbf{H}(p_i) \hat{\mathbf{n}} \tag{2.53}$$

Extending the error indicator proposed in [167, 113], a more complex error indicator is defined as [93]

$$e_i(p_i) = \frac{V_i^{2/d} |\hat{\boldsymbol{\tau}}^T \mathbf{H}(p_i) \hat{\boldsymbol{\tau}}|}{V_i^{1/d} |\hat{\boldsymbol{\tau}} \cdot \boldsymbol{\nabla} p_i| + \epsilon |\text{mean}(p)|} + \frac{V_i^{1/d} |\hat{\boldsymbol{\tau}} \cdot \boldsymbol{\nabla} p_i|}{V_i^{2/d} |\hat{\boldsymbol{\tau}}^T \mathbf{H}(p_i) \hat{\boldsymbol{\tau}}| + \epsilon |\text{mean}(p)|} \tag{2.54}$$

where the term $\epsilon|\text{mean}(p)|$ at denominator is used to avoid singularities.

In order to define a desired edge size map, the mean μ and the standard deviation σ of one of the above error indicators over the domain are defined. Then,

two refinement and coarsening thresholds are defined as

$$\begin{aligned} \tau_R &= \mu + k_R\sigma & \tau_C &= k_c\mu \\ \tau_{R_1} &= \mu + 2k_R\sigma & \tau_{C_1} &= \frac{k_c}{2}\mu \end{aligned} \tag{2.55}$$

where k_R, k_C are user-defined parameters. For each node in the mesh, once the current mean edge length is defined as

$$h_i = \frac{1}{N_i} \sum_{k \in \mathcal{K}_{i,\neq}} ||\mathbf{x}_i - \mathbf{x}_k|| \tag{2.56}$$

with N_i the number of edges connected to the node, a target mesh spacing (or metric map) $\bar{h}_i$ is defined as a discrete function of the error indicator as

$$\bar{h}_i(e_i) = \begin{cases} 0.25h_i & e_i \geq \tau_{R_1} \\ 0.5h_i & \tau_{R_1} > e_i \geq \tau_R \\ h_i & \tau_R > e_i > \tau_C \\ 2h_i & \tau_C \geq e_i > \tau_{C_1} \\ 4h_i & e_i \leq \tau_{C_1} \end{cases} \tag{2.57}$$

and finally the target spacing is required to fulfill some desired minimum $h_{\min}$ and maximum $h_{\max}$ edge size

$$\bar{h}_i = \min(\max(\bar{h}_i, h_{\min}), h_{\max}) \tag{2.58}$$

Following [2], a multi-passage strategy is implemented in order to allow the above metric map to capture simultaneous flow features of different strength. For instance, considering a shock wave and an expansion fan simultaneously present in the flow field, the statistics of the error indicator would be most likely dominated by the effects of the shock wave on the solution derivatives, thus preventing nodes in the expansion fan to be given a decreasing desired size. A sequential reapplication of the above procedure, with the exclusion of nodes already marked for refinement from the computation of the error indicator statistics, allows to assign a decreasing metric map also on nodes interested by less strong flow features.

Whenever it is necessary, the pointwise metric map (equation 2.57) is interpolated among neighbor nodes.

2.7.2 Anisotropic mesh adaptation

The isotropic metric map defined in section 2.7.1 is generalized to the anisotropic case by means of the error estimation analysis presented in [8, 66] and introduced in chapter 1.

As introduced in chapter 1, the desired metric tensor map $\mathscr{M}(\mathbf{x})$ is defined as a corrected eigendecomposition of the Hessian tensor of the solution

$$\tilde{\mathscr{M}} = \mathbf{R}\tilde{\Lambda}\mathbf{R}^{-1}, \qquad \tilde{\Lambda} = \begin{bmatrix} \tilde{\lambda}_1 & 0 & 0 \\ 0 & \tilde{\lambda}_2 & 0 \\ 0 & 0 & \tilde{\lambda}_3 \end{bmatrix} \tag{2.59}$$

$$\tilde{\lambda}_i = \min\left(\max\left(\frac{c_d|\lambda_i|}{\epsilon}, \frac{1}{h_{\max}^2}\right), \frac{1}{h_{\min}^2}\right) \tag{2.60}$$

with $\lambda_i, \mathbf{R}$ being the eigenvalues and eigenvectors of the Hessian tensor, $h_{\min}, h_{\max}$ being the minimum and maximum desired edge length, c_d a constant dependent on the space dimensions (9/32 in three dimensions) and ϵ being the maximum tolerated interpolation error.

2.8 Array-based data structures

The selection of a suitable data structure is essential for the feasibility and efficiency of mesh generation [63, 68, 65]. In the last `Flowmesh` version [140] implementing the link to the `MMG` library, an array-based data structure is employed. It is made of one pre-allocated array of structures (AoS) for each typology of mesh entity (elements, edges, nodes), where the structure in each array entry is allocated only if it corresponds to an existing mesh entity. A sufficiently large pre-allocation of the AoS allows for inserting additional mesh entities during adaptation without a significant memory penalty (since entity structure is allocated only when the entity is actually created in the mesh, and freed when it is deleted), while the array allows to preserve direct memory access. Each mesh entity structure stores both the downward connectivities (e.g. element-to-edges, element-to-nodes, edge-to-nodes ...) and the upward connectivities (e.g. node-to-elements...) to allow for graph search. The object-oriented implementation allows to *insert*, *delete*, *move*, *copy* mesh entities, while an additional *packing* operation is provided to restore contiguity of active array entries by moving entities from the bottom to the first entry of the array.

Part of this work has focused on completing the support to the above data structure in the solver code, thus removing conversions with legacy fixed-size array data structures. A major performance improvement has been obtained by adding an additional hash-based data structure for node-pairs, as it is shown in section 2.9.

2.9 Node-pair search by hashing

Other then inserting, moving, and deleting, *searching* mesh entities is an additional operation needed in many tasks [63, 68, 65]. This operation can become particularly challenging when we look for an entity given its downward connectivity, e.g. when looking for an element or an edge (or node-pair) given the nodes attached to it. Processing every entity and checking if its nodes correspond to those required would not be the most efficient solution. In this case, hashing is a possibility for storing node-pairs and index them based on their vertices [68].

Hashing is a form of database binning which, when applied to mesh edges, associates a single integer number (a *key*) to a pair of node indices. If the key is used as array index for the corresponding edge, this allows to query an edge given a pair of nodes with $\mathcal{O}(1)$ cost. *Collisions* are usually possible, i.e. different pairs of node indices returning the same key, which needs to be associated to several edges. The efficiency of hashing is retained as long as the number of collisions is low.

This strategy has been implemented for node-pair search, which is needed when an edge is destroyed and re-created during the same time step (as described in section 2.6). The employed hash table consists in a fixed-size array of structures (sufficiently large to contain the maximum number of keys envisioned for the problem) accessed by an integer key computed from a combination of the node indices. Before mesh adaptation, all the node-pairs are hashed (a linear cost operation) and their array index is stored in the hash table at the corresponding key. In case of collision, a single-link list is stored at the corresponding key. During mesh adaptation, the existence of a removed node-pair for a newly created edge can thus be checked with $\mathcal{O}(1)$ cost (instead of scanning, each time, the whole node-pair array looking for the good pair of nodes, whose cost grows as the product of the number of node-pairs and the number of created edges). Performance improvements are shown in table 2.1. The employed test case is the three-dimensional steady flow around the Onera M6 wing [144]. All tests are run on 12 CPUs, mesh adaptation is serial. Cases A, B, C refer to the three different initial meshes which have been employed for scaling purposes; cases B and C perform adaptation on the output mesh from cases A and B respectively. Mesh adaptation is isotropic, driven by the Hessian of the Mach number, with 3 levels of multi-passage technique performed with thresholds $k_r = 2.0$, $k_c = 0.18$, $h_{\text{grad}} = 1.38$ in the domain and $h_{\text{grad}} = 0.0005$ on the wing surface. Tests 1, 2, 3 refer to (1) the first software implementation, without hash tables and using `MMG` version 5.1.1, (2) the same software implementation, updated to the *develop* branch of `MMG` version 5.1.3 (which employs a fast octree search algorithm), (3) the final software implementation using both the develop branch of `MMG` version 5.1.3 and nodepair hash

Case	Mesh nodes	Mesh elements	Wall time	Adapt. time
$A1$	93432 → 165198	543590 → 942234	12h 59m 22s	5h 31m 46s
$A2$	93432 → 165213	543590 → 942551	9h 50m 24s	3h 58m 33s
$A3$	93432 → 165213	543590 → 942551	5h 11m 43s	15m 23s
$B1$	165198 → 290573	942234 → 1684211	32h 46m 02s	17h 39m 30s
$B2$	165198 → 268703	942234 → 1545451	16h 12m 16s	9h 7m 53s
$B3$	165198 → 268703	942234 → 1545451	7h 46m 7s	40m 9s
$C1$	290573 → 504768	1684211 → 2954999	86h 07m 33s	53h 20m 18s
$C2$	290573 → 444381	1684211 → 2586237	38h 13m 18s	22h 11m 9s
$C3$	290573 → 444381	1684211 → 2586237	17h 44m 16s	1h 18m 9s

Table 2.1 – Scaling tests on the Onera M6 wing, for different software implementations.

tables. Both total wall times and mesh adaptation times (in the `Flowmesh` coupling, thus including swept volumes computation) are presented in table 2.1. Performance improvement in adaptation times grows nonlinearly with mesh nodes, with 1.4×, 1.9×, 2.4× improvement factors due only to the update in the `MMG` version. Further improvements of 15.9×, 13.7×, 17.1× are achieved with the usage of nodepair hash tables, leading to total improvements in adaptation times of 22.1×, 26.5×, and 41×. With respect to total wall time, the time percentage spent in mesh adaptation improved from to 42.5%, 53.9%, 61.9% to 4.8%, 8.6%, 7.3% in the three test cases.

3 Numerical models for dynamic mesh adaptation with constant connectivity

To conclude this part on the mathematical and numerical aspects of conservative mesh adaptation for unsteady compressible flows, this chapter presents a complementary, self-contained work on the specific mesh mechanics algorithms for constant-connectivity mesh adaptation.

In this chapter, a Laplacian-based variational model for constant-connectivity mesh adaptation in the reference domain (inspired by [37, 38]) is extended to moving boundaries, and it is combined with an elasticity-based model inspired by [128] to provide a mixed model capable of overcoming some limitations of both its components.

The work shown in this chapter has been performed at the INRIA research center in Bordeaux, under the supervision of Cécile Dobrzynski and Mario Ricchiuto, with the `FMG` library for r-adaptation.

3.1 Introduction

Following the overview given in chapter 1.6, we consider PDE models for r-adaptation which can be obtained from the minimization of an adaptation functional. While most of these models are formulated for the parametric coordinates $\boldsymbol{\xi}(\mathbf{x})$ in the physical domain $\Omega_{\mathbf{x}}$, we follow the approach shown in [37] to directly formulate the problem for the mesh coordinates $\mathbf{x}(\boldsymbol{\xi})$ in the reference domain $\Omega_{\boldsymbol{\xi}}$, so that the resulting problem becomes similar to Lagrangian methods in computational mechanics.

In both the Laplacian-based and elasticity-based models which will be shown in the following, mesh adaptation is driven by a *monitor function* ω built from a scalar fluid flow solution $p(\mathbf{x})$ as

$$\omega(\mathbf{x}) = \sqrt{1 + \alpha ||\boldsymbol{\nabla}_{\boldsymbol{\xi}} p(\mathbf{x})||^2_{\gamma_\alpha} + \beta ||\mathbf{H}_{\boldsymbol{\xi}}(p)(\mathbf{x})||^2_{\gamma_\beta} + \tau ||p||^2_{\gamma_\tau}} \tag{3.1}$$

where $\boldsymbol{\nabla}_{\boldsymbol{\xi}}$ and $\mathbf{H}_{\boldsymbol{\xi}}$ denote the gradient and Hessian computed on the reference domain $\Omega_{\boldsymbol{\xi}}$, while their norm is defined as

$$||f||_\gamma = \min\left(1, \frac{||f||}{\gamma \max(||f||)}\right) \tag{3.2}$$

so that some saturation is added near the norm maximum according to the value of γ. The above definition allows the user to govern the intensity of mesh adaptation through the parameter pairs (α,γ_α), (β,γ_β), (τ,γ_τ).

3.2 Weak formulations

3.2.1 Laplacian model

Following [37], we formulate a Laplacian model for mesh adaptation in the reference domain $\Omega_{\boldsymbol{\xi}}$

$$\boldsymbol{\nabla}_{\boldsymbol{\xi}} \cdot \left(\omega(\mathbf{x})\boldsymbol{\nabla}_{\boldsymbol{\xi}}\mathbf{x}\right) = \mathbf{0} \qquad \text{in } \Omega_{\boldsymbol{\xi}} \tag{3.3}$$

whose boundary is split as $\partial\Omega_{\boldsymbol{\xi}} = \Gamma^D_{\boldsymbol{\xi}} \cup \Gamma^S_{\boldsymbol{\xi}}$ so that Dirichlet conditions are imposed on $\Gamma^D_{\boldsymbol{\xi}}$ and slip conditions are imposed on $\Gamma^S_{\boldsymbol{\xi}}$

$$\begin{aligned} \mathbf{x} &= \boldsymbol{\xi} && \text{on } \Gamma^D_{\boldsymbol{\xi}} \\ \hat{\mathbf{n}} \cdot (\mathbf{x} - \boldsymbol{\xi}) &= 0 && \text{on } \Gamma^S_{\boldsymbol{\xi}} \end{aligned} \tag{3.4}$$

With the imposition of an additional Neumann condition ,$\hat{\mathbf{n}} \cdot \boldsymbol{\nabla}_{\boldsymbol{\xi}}\mathbf{x} = \mathbf{0}$ on $\Gamma^S_{\boldsymbol{\xi}}$, a variational formulation for the above problem can be readily obtained as

$$\int_{\Omega_\xi} \omega(\mathbf{x})\boldsymbol{\nabla}_{\boldsymbol{\xi}} v \cdot \boldsymbol{\nabla}_{\boldsymbol{\xi}}\mathbf{x}\, \mathrm{d}\Omega_{\boldsymbol{\xi}} = \mathbf{0}, \quad \forall v \in H^1(\Omega_{\boldsymbol{\xi}}) \tag{3.5}$$

For each space direction, the above equations are uncoupled and nonlinear through the monitor function $\omega(\mathbf{x})$, which satisfy the double role of driving mesh adaptation towards the flow patterns captured by the monitor function, and of increasing element stiffness. This can be better appreciated by defining the displacements

$$\boldsymbol{\delta} = \mathbf{x} - \boldsymbol{\xi} \tag{3.6}$$

and inserting them in the Laplace equation 3.3, to get

$$\boldsymbol{\nabla}_{\boldsymbol{\xi}} \cdot \left(\omega\boldsymbol{\nabla}_{\boldsymbol{\xi}}\boldsymbol{\delta}\right) = -\boldsymbol{\nabla}_{\boldsymbol{\xi}}\omega \qquad \text{in } \Omega_{\boldsymbol{\xi}} \tag{3.7}$$

Thus, mesh nodes displacements are driven by the gradients of the monitor function, which also provides a variable stiffness term.

3.2.2 Elastic model

We consider the domain as an elastic solid whose displacements $\boldsymbol{\delta}$ are governed by the equilibrium equations [135]

$$\boldsymbol{\nabla}_{\boldsymbol{\xi}} \cdot \boldsymbol{\sigma}(\boldsymbol{\delta}) + \mathbf{f} = \mathbf{0} \qquad \text{in } \Omega_{\boldsymbol{\xi}} \tag{3.8}$$

where $\boldsymbol{\sigma}$ is the Cauchy stress tensor, and $\mathbf{f}$ is a volume force, whose definition is inspired by the Laplacian model

$$\mathbf{f} = \boldsymbol{\nabla}_{\boldsymbol{\xi}} \omega(\mathbf{x}) \tag{3.9}$$

The domain boundary is split as $\partial\Omega_{\boldsymbol{\xi}} = \Gamma^D_{\boldsymbol{\xi}} \cup \Gamma^S_{\boldsymbol{\xi}}$ so that Dirichlet conditions are imposed on $\Gamma^D_{\boldsymbol{\xi}}$ and slip conditions are imposed on $\Gamma^S_{\boldsymbol{\xi}}$

$$\begin{aligned} \boldsymbol{\delta} &= \mathbf{0} && \text{on } \Gamma^D_{\boldsymbol{\xi}} \\ \hat{\mathbf{n}} \cdot \boldsymbol{\delta} &= 0 && \text{on } \Gamma^S_{\boldsymbol{\xi}} \end{aligned} \tag{3.10}$$

A small displacement hypothesis is introduced so that the small deformation tensor is defined as

$$\boldsymbol{\epsilon}(\boldsymbol{\delta}) = \frac{1}{2}\left(\boldsymbol{\nabla}_{\boldsymbol{\xi}} \boldsymbol{\delta} + (\boldsymbol{\nabla}_{\boldsymbol{\xi}} \boldsymbol{\delta})^T\right) \tag{3.11}$$

leading to a variational formulation

$$\int_{\Omega_{\boldsymbol{\xi}}} \boldsymbol{\epsilon}(\mathbf{v}) : \boldsymbol{\sigma}(\boldsymbol{\delta}) \, \mathrm{d}\Omega_{\boldsymbol{\xi}} = \int_{\Omega_{\boldsymbol{\xi}}} \mathbf{v} \cdot \boldsymbol{\nabla} \omega(\mathbf{x}) \, \mathrm{d}\Omega_{\boldsymbol{\xi}}, \qquad \forall \mathbf{v} \in [H^1_0(\Omega_{\boldsymbol{\xi}})]^d \tag{3.12}$$

A linear constitutive equation is used

$$\boldsymbol{\sigma}(\boldsymbol{\delta}) = 2\mu \boldsymbol{\epsilon}(\boldsymbol{\delta}) + \lambda \mathrm{tr}\left(\boldsymbol{\epsilon}(\boldsymbol{\delta})\right) \mathbf{I} \tag{3.13}$$

With respect to the Laplacian model, the elastic one is linear, allowing for a single evaluation of the system matrix, and equations are coupled in different space directions. The choice of constant Lamé coefficients μ, λ doesn't allow to exploit the automatic stiffening effects which is present in the Laplacian model, so care is needed to avoid mesh tangling for high values of forcing.

3.3 $\mathbb{P}_1$ finite element discretization

Introducing a linear finite element basis, gradients become constant functions that can be analytically evaluated through geometric considerations. On a generic element k, the gradient of a nodal basis function ϕ_i at node i can be expressed as a function of the outward unit vector $\hat{\mathbf{n}}_i$ orthogonal to the face opposed to node i and its height h_i as

$$\boldsymbol{\nabla}_{\boldsymbol{\xi}} \phi_i = -\frac{1}{h_i} \hat{\mathbf{n}}_i \tag{3.14}$$

Height h_i is readily obtained from the relation between the element volume $|\Omega_k|$ and the face area $|F_i|$

$$|\Omega_k| = \frac{|F_i| h_i}{d} \tag{3.15}$$

so that the gradient can be written as

$$\boldsymbol{\nabla}_{\boldsymbol{\xi}}\phi_i = -\frac{|F_i|\hat{\mathbf{n}}_i}{d|\Omega_k|} \tag{3.16}$$

Defining the integrated normal vector[1]

$$\mathbf{n}_i = -(d-1)!|F_i|\hat{\mathbf{n}}_i \tag{3.18}$$

the gradient is finally expressed as

$$\boldsymbol{\nabla}_{\boldsymbol{\xi}}\phi_i = \frac{\mathbf{n}_i}{d!|\Omega_k|} \tag{3.19}$$

The last expression will be used to compute gradients appearing in the variational formulation presented in the previous sections.

3.3.1 Laplacian model

Once a $\mathbb{P}_1$ basis function is chosen, each space coordinate $x^k, k = 1,\dots,d$ is expanded as $x^d = \sum_{j=1}^{n_p} \phi_j x_j^k$ so that the Galerkin projection reads

$$\sum_{j=1}^{n_p} \int_{\Omega_{\boldsymbol{\xi}}} \omega(\mathbf{x}) \boldsymbol{\nabla}_{\boldsymbol{\xi}}\phi_i \cdot \boldsymbol{\nabla}_{\boldsymbol{\xi}}\phi_j \, \mathrm{d}\Omega_{\boldsymbol{\xi}} x_j^k = 0, \quad \forall i = 1,\dots,n_p \tag{3.20}$$

The monitor function on each element will be considered constant and equal to the mean of the nodal value on the element, so that by inserting the gradient expression 3.19 leads to the definition of the stiffness matrix entries

$$K_{ij}(\mathbf{x}) = \int_{\Omega_{\boldsymbol{\xi}}} \omega(\mathbf{x}) \boldsymbol{\nabla}_{\boldsymbol{\xi}}\phi_i \cdot \boldsymbol{\nabla}_{\boldsymbol{\xi}}\phi_j \, \mathrm{d}\Omega_{\boldsymbol{\xi}} = \sum_{k \in \mathscr{B}_i \cap \mathscr{B}_j} \bar{\omega}_k(\mathbf{x}) \frac{\mathbf{n}_i \cdot \mathbf{n}_j}{(d!)^2 |\Omega_k|} \tag{3.21}$$

3.3.2 Elastic model

Introducing a linear nodal finite element basis, we would like to write the stiffness matrix entries for nodes h, k, coupling space directions i, l, as a generalization of the Laplacian case (eq. 3.21)

$$K_{hk}^{il} = \int_{\Omega_{\boldsymbol{\xi}}} \left(\boldsymbol{\nabla}_{\boldsymbol{\xi}}\phi_h\right)^T \mathbf{A}^{il} \boldsymbol{\nabla}_{\boldsymbol{\xi}}\phi_k \, \mathrm{d}\Omega_{\boldsymbol{\xi}} = \sum_{s \in \mathscr{B}_h \cap \mathscr{B}_k} \frac{\mathbf{n}_h^T \mathbf{A}^{il} \mathbf{n}_k}{(d!)^2 |\Omega_s|} \tag{3.22}$$

[1] While in $d = 2$ dimensions the factorial term gives no real contribution to the above definition, in $d = 3$ dimensions the term $(d-1)! = 2$ it allows to directly compute the integrated normal vector $\mathbf{n}_i$ as the oriented area of the parallelogram formed by two arbitrary edge vectors $\mathbf{e}_{01}, \mathbf{e}_{02}$ on the face F_i (provided that its nodes 0,1,2 have counter-clockwise orientation) as

$$\mathbf{n}_i = \mathbf{e}_{02} \times \mathbf{e}_{01} \tag{3.17}$$

Denoting displacements as $\mathbf{u} = \boldsymbol{\delta}$ to avoid confusion, thanks to the symmetry of the stress tensor, the double product in the variational formulation is compactly re-expressed as

$$\int_{\Omega_{\boldsymbol{\xi}}} \boldsymbol{\epsilon}(\mathbf{v}) : \boldsymbol{\sigma}(\mathbf{u}) \, \mathrm{d}\Omega_{\boldsymbol{\xi}} = \int_{\Omega_{\boldsymbol{\xi}}} \boldsymbol{\nabla}_{\boldsymbol{\xi}} \mathbf{v} : \boldsymbol{\sigma}(\mathbf{u}) \, \mathrm{d}\Omega_{\boldsymbol{\xi}} \tag{3.23}$$

Using Einstein's notation

$$\boldsymbol{\nabla}_{\boldsymbol{\xi}} \mathbf{v} : \boldsymbol{\sigma}(\mathbf{u}) = v_{i,j} \sigma_{ij} \tag{3.24}$$

we aim at rewriting the double product as a function of basis function gradients $\phi^k_{,m}$, as in the Laplacian case, by looking for a relation of the kind

$$v^h_i \phi^h_{,j} A^{il}_{jm} \phi^k_{,m} u^k_l \tag{3.25}$$

so that the stiffness matrix entry for nodes h, k, coupling space directions i, l, will read

$$K^{il}_{hk} = \int_{\Omega_{\boldsymbol{\xi}}} \phi^h_{,j} A^{il}_{jm} \phi^k_{,m} \, \mathrm{d}\Omega_{\boldsymbol{\xi}} \tag{3.26}$$

The above relation can be obtained by introducing the constitutive relation for the Cauchy stress tensor, and the definition of the small strain tensor, into the double product

$$\begin{aligned} v_{i,j}\sigma_{ij} &= v_{i,j}\left(\mu\epsilon_{ij} + \lambda\epsilon_{mm}\delta_{ij}\right) = \\ &= v^h_i \phi^h_{,j}\left(\mu\left(\frac{1}{2}\phi^k_j u^k_i + \frac{1}{2}\phi^k_i u^k_j\right) + \lambda\epsilon_{mm}\right) = \\ &= v^h_i \phi^h_{,j}\left(\frac{1}{2}\mu\phi^k_j u^k_i + \frac{1}{2}\mu\phi^k_i u^k_j + \lambda\phi^k_{,m} u^k_m \delta_{ij}\right) = \\ &= v^h_i \phi^h_{,j}\left(\frac{1}{2}\mu\delta_{il}\delta_{jm} + \frac{1}{2}\mu\delta_{im}\delta_{jl} + \lambda\delta_{ij}\delta_{lm}\right)\phi^k_{,m} u^k_l \end{aligned} \tag{3.27}$$

For each couple of indices i, l a matrice with indices j, m can be written

$$A^{il}_{jm} = \underbrace{\frac{1}{2}\mu\delta_{il}\delta_{jm}}_{\text{4D diagonal}} + \underbrace{\frac{1}{2}\mu\delta_{im}\delta_{jl}}_{\text{Entry } li} + \underbrace{\lambda\delta_{ij}\delta_{lm}}_{\text{Entry } il} \tag{3.28}$$

Since $\mathbf{A}^{il}$ is symmetric, in three dimensions the six independent matrices read

$$\begin{aligned} \mathbf{A}^{11} &= \begin{bmatrix} \mu+\lambda & 0 & 0 \\ 0 & \frac{1}{2}\mu & 0 \\ 0 & 0 & \frac{1}{2}\mu \end{bmatrix} \quad \mathbf{A}^{12} = \begin{bmatrix} 0 & \lambda & 0 \\ \frac{1}{2}\mu & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \\ \mathbf{A}^{22} &= \begin{bmatrix} \frac{1}{2}\mu & 0 & 0 \\ 0 & \mu+\lambda & 0 \\ 0 & 0 & \frac{1}{2}\mu \end{bmatrix} \quad \mathbf{A}^{23} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \lambda \\ 0 & \frac{1}{2}\mu & 0 \end{bmatrix} \\ \mathbf{A}^{33} &= \begin{bmatrix} \frac{1}{2}\mu & 0 & 0 \\ 0 & \frac{1}{2}\mu & 0 \\ 0 & 0 & \mu+\lambda \end{bmatrix} \quad \mathbf{A}^{31} = \begin{bmatrix} 0 & 0 & \frac{1}{2}\mu \\ 0 & 0 & 0 \\ \lambda & 0 & 0 \end{bmatrix} \end{aligned} \tag{3.29}$$

3.4 Jacobi iterative solution

The algebraic systems derived in the previous section are solved by means of a Jacobi iterative method [139]. The procedure is detailed here for both the Laplacian and the elastic model.

3.4.1 Laplacian model

Compactly denoting the array of unknown node positions $\mathbf{x} = [x_i^k]$ and the system matrix $\mathbf{K}$, the nonlinear algebraic system for the Laplacian model reads

$$\mathbf{K}(\mathbf{x})\mathbf{x} = \mathbf{0} \tag{3.30}$$

Introducing again the displacement $\boldsymbol{\delta} = \mathbf{x} - \boldsymbol{\xi}$, the system is rewritten as

$$\mathbf{K}(\mathbf{x})\boldsymbol{\delta} = -\mathbf{K}(\mathbf{x})\boldsymbol{\xi} \tag{3.31}$$

Jacobi iterations allow to uncouple each nodal position $\mathbf{x}_i, i = 1,\dots,N_p$ by splitting the matrix into the diagonal and extra-diagonal parts

$$k_{ii}\boldsymbol{\delta}_i = -\sum_{\substack{j\neq i \\ j\in\mathscr{B}_i}} k_{ij}\boldsymbol{\delta}_j - \sum_{j\in\mathscr{B}_i} k_{ij}\boldsymbol{\xi}_j \tag{3.32}$$

and then evaluating the left-hand-side at the current iteration $k+1$, and the right-hand-side at the previous iteration k to get an iteration equation

$$k_{ii}^{[k]}\boldsymbol{\delta}_i^{[k+1]} = -\sum_{\substack{j\neq i \\ j\in\mathscr{B}_i}} k_{ij}^{[k]}\boldsymbol{\delta}_j^{[k]} - \sum_{j\in\mathscr{B}_i} k_{ij}^{[k]}\boldsymbol{\xi}_j \tag{3.33}$$

Adding the term $k_{ii}^{[k]}\boldsymbol{\delta}_j^{[k]}$ to both sides allows to finally write a simpler equation

$$k_{ii}^{[k]}\boldsymbol{\delta}_i^{[k+1]} = k_{ii}^{[k]}\boldsymbol{\delta}_j^{[k]} - \sum_{j\in\mathscr{B}_i} k_{ij}^{[k]}\mathbf{x}_j^{[k]} \tag{3.34}$$

Thus, after the initialization $\boldsymbol{\delta}_i^{[0]} = \mathbf{0}$, Jacobi iterations for $k = 1,\dots,K$ are performed as

$$\begin{aligned} \boldsymbol{\delta}_i^{[k+1]} &= \boldsymbol{\delta}_i^{[k]} - \frac{1}{k_{ii}^{[k]}} \sum_{j\in\mathscr{B}_i} k_{ij}^{[k]}\mathbf{x}_j^{[k]} \\ \mathbf{x}_i^{[k+1]} &= \mathbf{x}_i^{[k]} + \theta\left(\boldsymbol{\xi}_i + \boldsymbol{\delta}_i^{[k+1]} - \mathbf{x}_i^{[k]}\right) \end{aligned} \tag{3.35}$$

where $\theta \in [0,1]$ is a relaxation parameter, allowing to reduce the imposed displacement with respect to the previous iteration, in case it produces invalid elements.

3.4.2 Elastic model

Similarly to the Laplacian case, the algebraic system for the elasticity equations reads

$$\mathbf{K}\boldsymbol{\delta} = \mathbf{b}(\mathbf{x}) \tag{3.36}$$

where $\mathbf{b}$ is the discretization of the forcing term, dependent on position through the monitor function $\omega(\mathbf{x})$. As before, the system matrix is split into a diagonal and extra-diagonal part

$$k_{ii}\boldsymbol{\delta}_i = \mathbf{b} - \sum_{\substack{j\neq i \\ j\in\mathscr{B}_i}} k_{ij}\boldsymbol{\delta}_j \tag{3.37}$$

then left and right hand sides are evaluated at different steps in order to obtain the Jacobi iteration for each node i

$$\begin{aligned} \boldsymbol{\delta}_i^{[k+1]} &= \frac{1}{k_{ii}}\left(\mathbf{b}^{[k]} - \sum_{\substack{j\neq i \\ j\in\mathscr{B}_i}} k_{ij}\boldsymbol{\delta}_j^{[k]}\right) \\ \mathbf{x}_i^{[k+1]} &= \mathbf{x}_i^{[k]} + \theta\left(\boldsymbol{\xi}_i + \boldsymbol{\delta}_i^{[k+1]} - \mathbf{x}_i^{[k]}\right) \end{aligned} \tag{3.38}$$

where the same elements validity check is performed for the computation of the relaxation parameter θ.

In order to improve the convergence of the iterative method, we introduce a parameter $\sigma \geq 0$ to modify the split equation 3.37 as follows

$$(k_{ii}+\sigma)\,\boldsymbol{\delta}_i = \mathbf{b} + \sigma\boldsymbol{\delta}_i - \sum_{\substack{j\neq i \\ j\in\mathscr{B}_i}} k_{ij}\boldsymbol{\delta}_j \tag{3.39}$$

In this way, by evaluating the left-hand-side at iteration $k+1$ and the right-hand-side at iteration k, we obtain the iteration equation

$$\boldsymbol{\delta}_i^{[k+1]} = \frac{1}{k_{ii}+\sigma}\left(\mathbf{b}^{[k]} + \sigma\boldsymbol{\delta}_i^{[k]} - \sum_{\substack{j\neq i \\ j\in\mathscr{B}_i}} k_{ij}\boldsymbol{\delta}_j^{[k]}\right) \tag{3.40}$$

which is in fact an under-relaxed version of the Jacobi method [2].

[2] By defining an equivalent parameter ω as

$$\omega = \frac{k_{ii}}{k_{ii}+\sigma} \tag{3.41}$$

equation 3.40 can be recast in the conventional form of a Jacobi Over (Under) Relaxation (JOR) method [139]

$$\boldsymbol{\delta}_i^{[k+1]} = \frac{\omega}{k_{ii}}\left(\mathbf{b}^{[k]} - \sum_{\substack{j\neq i \\ j\in\mathscr{B}_i}} k_{ij}\boldsymbol{\delta}_j^{[k]}\right) + (1-\omega)\boldsymbol{\delta}_i^{[k]} \tag{3.42}$$

3.5 Mixed model

In order to overcome some limitations of the Laplacian and the linear elastic models, namely the nonlinearity of the first (with the compulsory re-computation of the system matrix at each Jacobi iteration) and the inability of the latter to withstand strong solution gradients (due to its constant stiffness coefficients), a procedure for blending the two of them have been introduced. For each iteration k of the Laplacian model, we run $m = 1,\dots,M$ iterations of the elastic model. Finally, the displacements $\boldsymbol{\delta}_{i,L}^{[k+1]}$ given by the Laplacian model and the displacements $\boldsymbol{\delta}_{i,E}^{[k+1]}$ given by the elastic model are mixed and the new positions $\mathbf{x}_i^{[k+1]}$ are computed according to the relations

$$\begin{aligned}\boldsymbol{\delta}_i^{[k+1]} &= \left(1-b(p)\right)\boldsymbol{\delta}_{i,L}^{[k+1]} + b(p)\boldsymbol{\delta}_{i,E}^{[k+1,M]} \\ \mathbf{x}_i^{[k+1]} &= \mathbf{x}_i^{[k]} + \theta\left(\boldsymbol{\xi}_i + \boldsymbol{\delta}_i^{[k+1]} - \mathbf{x}_i^{[k]}\right)\end{aligned} \tag{3.43}$$

The scalar blending function $b:\mathbb{R}\rightarrow[0,1]$ is defined as

$$b(p) = \frac{h(p)-h_{\min}}{h_{\max}-h_{\min}} \tag{3.44}$$

where $h_{\min}, h_{\max}$ are the minimum and maximum prescribed edge sizes, and $h(p)$ is an Hessian-based edge size map inspired by anisotropic mesh adaptation [66] and defined as

$$h(p) = \frac{1}{\sqrt{\min(\max\left(\frac{c_d}{\epsilon}\lambda_{\max}\left(\mathbf{H}_{\boldsymbol{\xi}}(p)\right), h_{\max}^{-2}\right), h_{\min}^{-2})}} \tag{3.45}$$

where $\lambda_{\max}$ represent the maximum absolute value of the eigenvalues of the solution Hessian. Equation 3.45 can be rewritten as

$$h(p) = \max\left(\min\left(\frac{\epsilon^{1/2}}{c_d^{1/2}}\frac{1}{\lambda_{\max}^{1/2}(p)}, h_{\max}\right), h_{\min}\right) \tag{3.46}$$

and it allows to distribute the Laplacian and the elastic models so that the Laplacian one is used near the strongest solution variations, while the elastic one is employed in the smoothest flow regions [3].

[3] Given the expression of the edge size map (equation 3.45), the blending function (equation 3.44) depends from the scalar flow solution p and from parameter ϵ as follows

$$b(p;\epsilon) = \begin{cases} 1 & \epsilon \geq c_d\lambda_{max}(p)h_{max}^2 \\ \frac{\epsilon^{1/2}-c_d^{1/2}\lambda_{\max}^{1/2}(p)h_{min}}{c_d^{1/2}\lambda_{\max}^{1/2}(p)(h_{max}-h_{min})} & c_d\lambda_{max}(p)h_{min}^2 < \epsilon < c_d\lambda_{max}(p)h_{max}^2 \\ 0 & \epsilon \leq c_d\lambda_{max}(p)h_{min}^2 \end{cases} \tag{3.47}$$

3.6 Dynamic mesh adaptation

Following [154, 38], dynamic mesh adaptation during the time evolution of a fluid flow simulation is performed by repeating the steady adaptation procedure described in the previous section at each time step, without the explicit formulation of a differential equation in time for mesh motion, thus greatly simplifying coupling with existing flow solvers.

Unsteady flows over fixed boundary domains. In this case, the reference mesh $\boldsymbol{\xi}$ is constant in time, while the computational mesh $\mathbf{x}(t^{(n+1)})$ is the r-adaptation of the (fixed) reference mesh. Thus, the displacement at each time step $n+1$ is initialized with the value achieved at the last Jacobi iteration K achieved in the previous time step n

$$\boldsymbol{\delta}_i^{[0](n+1)} = \boldsymbol{\delta}_i^{[K](n)} \tag{3.48}$$

so that successive Jacobi iterations during time evolution are effectively accumulated on the nodes position

$$\mathbf{x}_i^{[k](n)} = \boldsymbol{\delta}_i^{[k](n)} + \boldsymbol{\xi}_i, \qquad k = 1,\dots,K, \quad n = 0,1,2,\dots \tag{3.49}$$

At each time step, the flow solution is predicted on the previous computational mesh, then the computational mesh is adapted, and finally the flow solution is recomputed on the adapted mesh.

Unsteady flow over moving boundary domains. In this case, the reference mesh $\boldsymbol{\xi}(t^{(n+1)})$ evolves in time (due to the need to comply with moving boundaries), and the computational mesh $\mathbf{x}(t^{(n+1)})$ is the r-adaptation of the reference mesh at the current time. Differently from the fixed boundary case, the evolution of the reference mesh prevents the direct accumulation of Jacobi iteration in the mesh nodes position. Thus, at each time step the reference mesh is deformed to comply with boundary motion, then the flow solution is predicted on the deformed reference mesh, and finally the computational mesh is adapted and the flow solution is recomputed on the adapted mesh.

3.7 Model assessment on an analytical function in 2D

The mesh adaptation models are tested in this section on an analytical solution function

$$p = e^{\theta\psi^2}, \qquad \psi = \sqrt{x^2+y^2} - R \tag{3.50}$$

with $R = 0.5$, in a square domain $[-2,2] \times [-2,2]$, on a uniform mesh with 7636 triangular elements. This function is chosen in order to test capability of the models

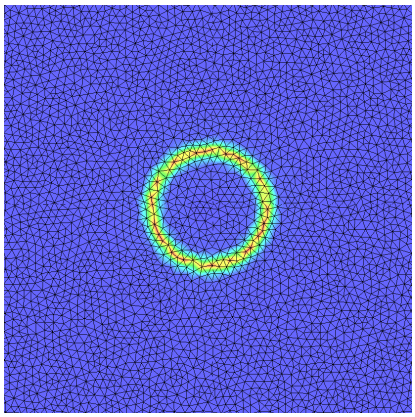
to adapt on a circle represented by a smooth solution field, before their application to solutions with sharp features, like shock waves. Figure 3.1 shows the effects of the Laplacian and the elastic models. Both models are run with the adaptation parameter values $\alpha = 0$, $\beta = 0$, $\tau = 100$, $\gamma_\tau = 1$ for the definition of the monitor function ω (eq 3.1), but different values of θ (400 for the Laplacian, 40 for the elasticity) are taken in order to avoid avoid mesh tangling for the elastic model. Results are presented for 20 Jacobi iterations (a typical value in unsteady flow simulations), and in the limit of many Jacobi iterations (3000 for the Laplacian model, 180 for the elastic model). Some considerations can be made.

- Given the same solution feature (in this case, a circle), the elastic model needs a much smoother monitor function with respect to the Laplacian model to produce valid meshes. This property becomes essential for application on shock waves, and it prevents in practice the application of a purely linear elastic model to transonic and supersonic flow solutions.

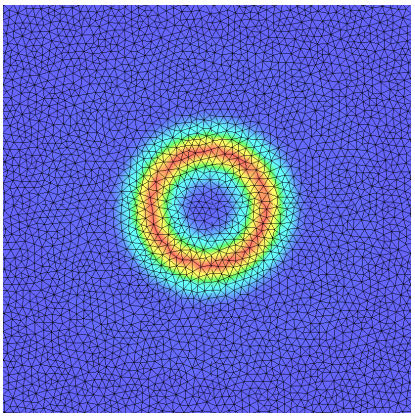
- The Laplacian model has a strong local effect, i.e. it is extremely capable to refine on sharp solution fronts while preserving good quality in small elements, but deformation does not propagate quickly in the domain, producing a strong element stretching immediately after and before the solution front.

- In the limit of many Jacobi iterations, the Laplacian model is still able to preserve good quality in the refined zones (for as mush as 3000 iterations, in this test case), while the elastic model fails in this respect (in this case, limiting the allowed iterations to 180), but it is able to produce much smoother meshes far from the solution fronts (zoom in figure 3.2).

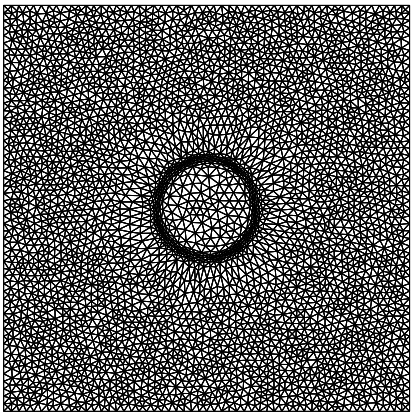
The main motivation for the mixed model developed in section 3.5 is to preserve the good element quality in the refined zones produced by the Laplacian model, while maintaining a smooth mesh outside solution fronts, as in the elastic case. Figure 3.3 shows the results for the mixed model applied to the same analytical solution function, for $\theta = 400$, with Laplacian parameter values $\alpha = 0$, $\beta = 0$, $\tau = 100$, $\gamma_\tau = 1$, and elasticity parameter values $\alpha = 0$, $\beta = 10$, $\gamma_\beta = 0.1$, $\tau = 100$, $\gamma_\tau = 1$ varying the value of the parameter ϵ, which governs the blending of Laplacian and elastic model, for 20 Jacobi iterations and in the limit of many iterations. For each Jacobi iterations, 200 iterations of the elastic model are performed. The smoothening effect given by elasticity is visible in the smooth regions away from the solution front, especially inside the circle. As the value of ϵ is increased, the importance of the elastic part increases and the robustness of the model is reduced, as can be seen from the instability of the refined circular zone and the decreasing



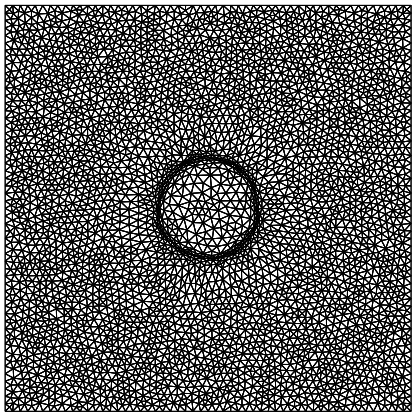
a Analytical function (eq. 3.50) on the initial mesh, for $\theta = 400$.



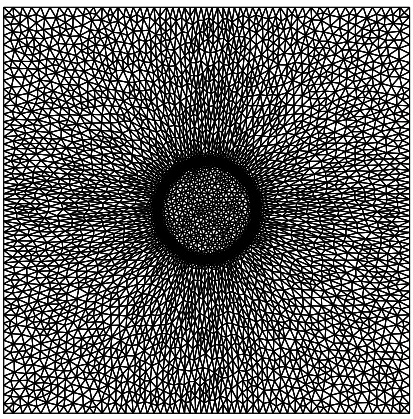
b Analytical function (eq. 3.50) on the initial mesh, for $\theta = 40$.



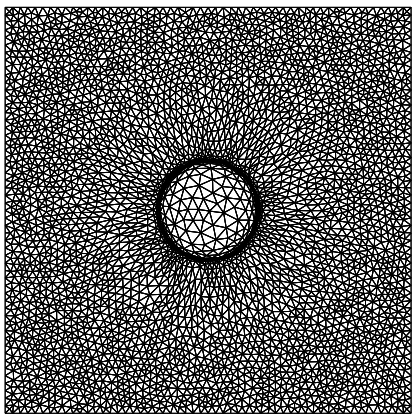
c Adaptation with Laplacian model, $\theta = 400$, 20 Jacobi iterations.



d Adaptation with elasticity model, $\theta = 40$, 20 Jacobi iterations.

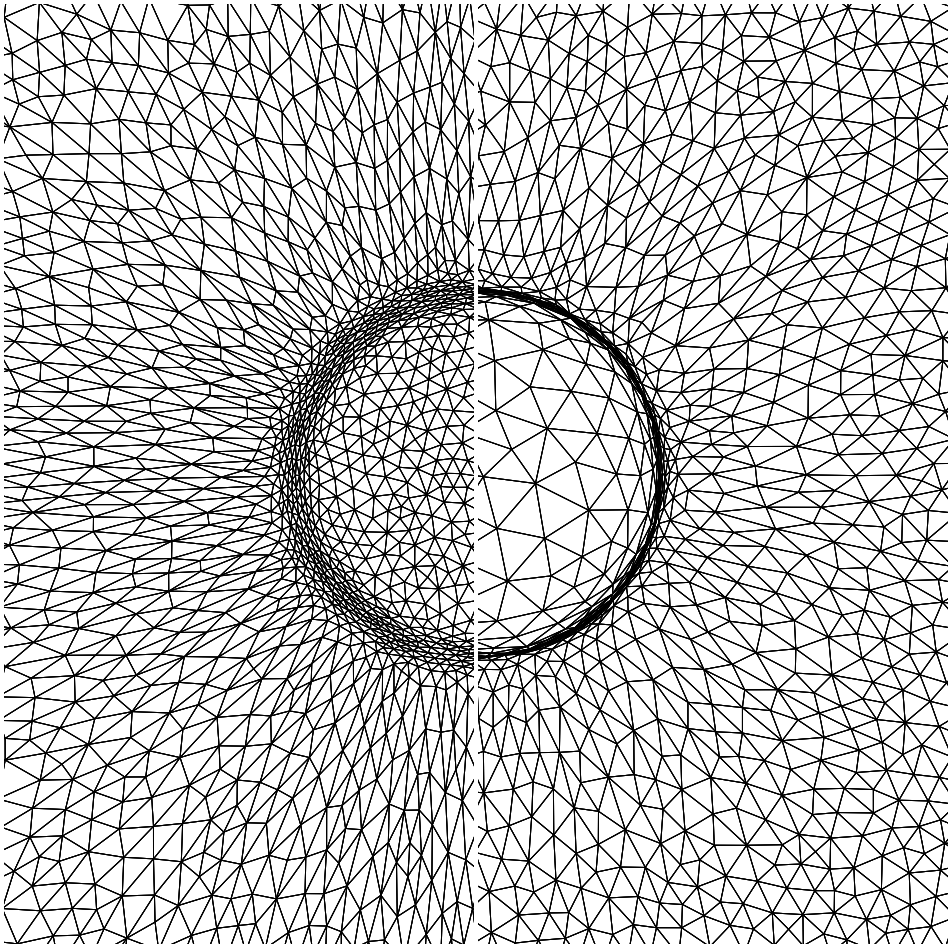


e Adaptation with Laplacian model, $\theta = 400$, 3000 Jacobi iterations.



f Adaptation with elasticity model, $\theta = 40$, 180 Jacobi iterations.

Figure 3.1 – Adaptation on analytical solution functions in two dimensions, Laplacian and elastic models.



(a) Laplacian, $\theta = 400$, 3000 Jacobi iterations.(b) Elasticity, $\theta = 40$, 180 Jacobi iterations.

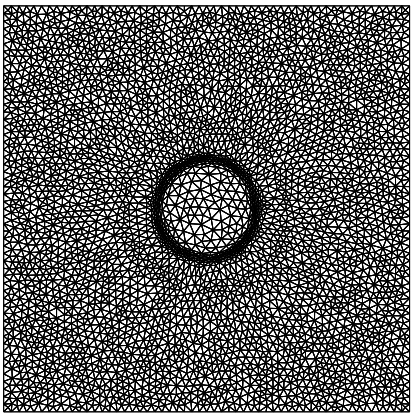
Figure 3.2 – Zoom on Laplacian (left) and elastic (right) model adaptation for many Jacobi iterations.

maximum number of Jacobi iteration that is possible to perform before incurring in mesh tangling. Figure 3.4 shows a comparison between Laplacian and mixed model, for $\epsilon = 10^{-3}$ and 20 Jacobi iterations. Refinement on the circular front appears to be preserved, while the element stretching immediately outside of it is significantly reduced.

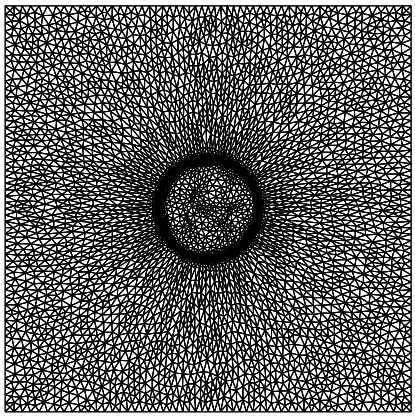
3.8 Model assessment on an analytical function in 3D

Assessment of the capabilities of the model in three-dimensions, with curved boundaries, is performed by using the analytical solution function

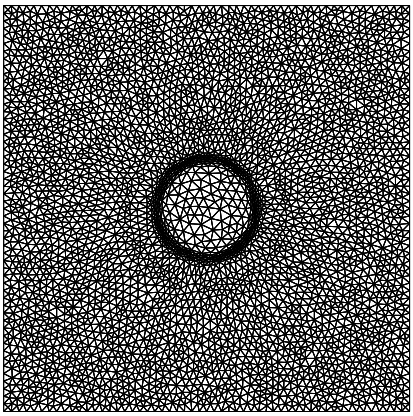
$$p = e^{-100(y-x^2-z^2)^2} \tag{3.51}$$



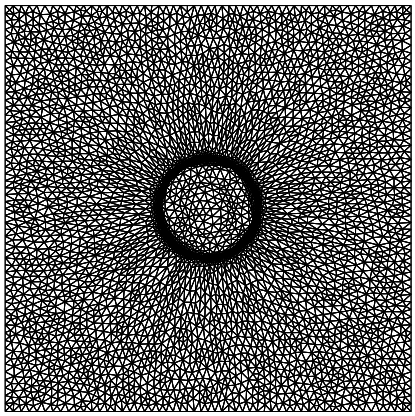
a Mixed model, $\epsilon = 5 \times 10^{-4}$, 20 Jacobi iterations.



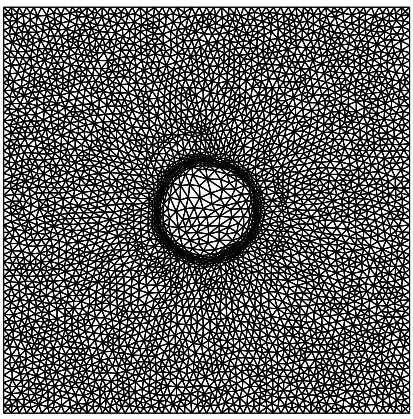
b Mixed model, $\epsilon = 5 \times 10^{-4}$, 651 Jacobi iterations.



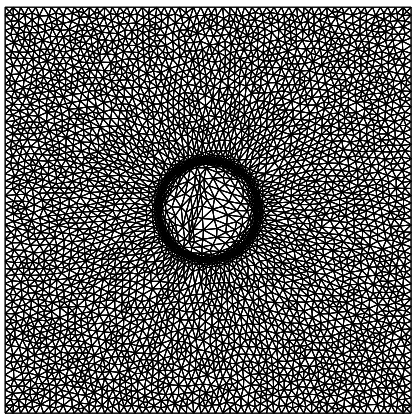
c Mixed model, $\epsilon = 10^{-3}$, 20 Jacobi iterations.



d Mixed model, $\epsilon = 10^{-3}$, 270 Jacobi iterations.

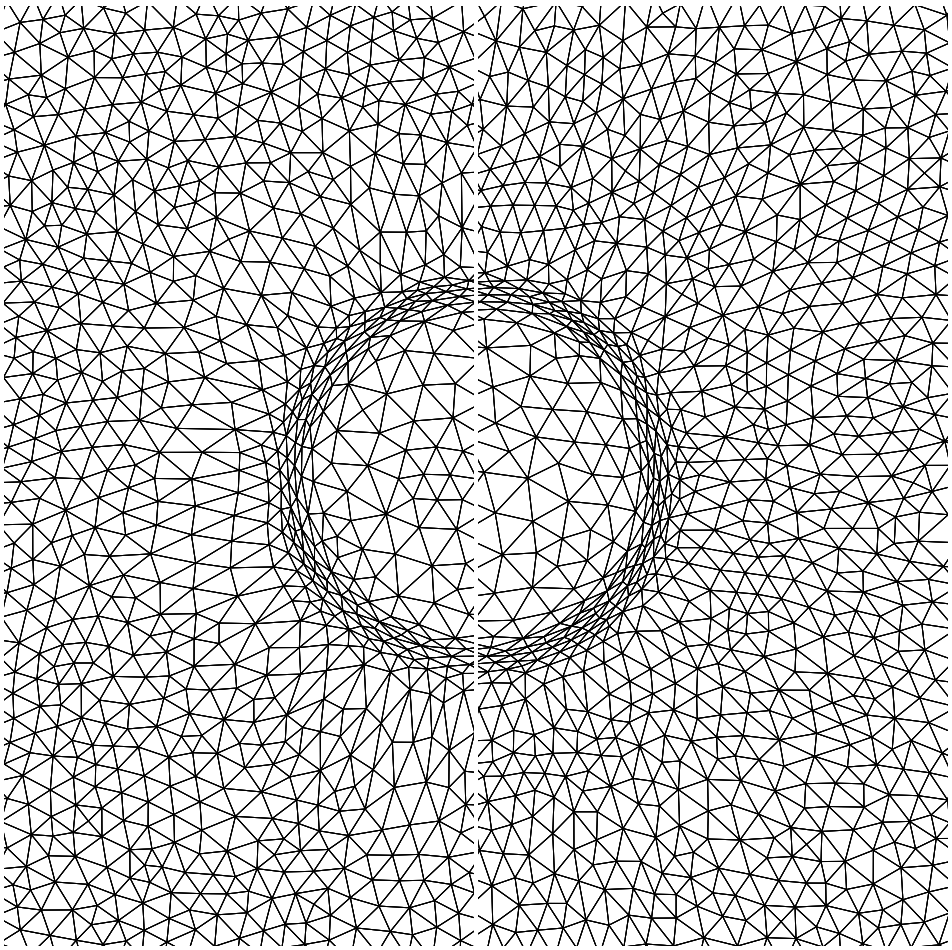


e Mixed model, $\epsilon = 5 \times 10^{-3}$, 20 Jacobi iterations.



f Mixed model, $\epsilon = 5 \times 10^{-5}$, 77 Jacobi iterations.

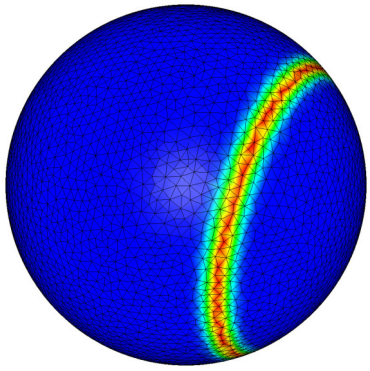
Figure 3.3 – Adaptation with the mixed model, for varying values of ϵ.



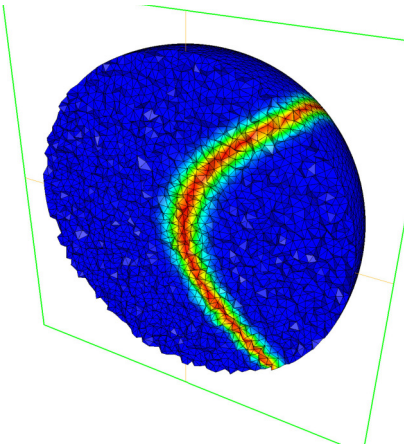
(a) Laplacian model. (b) Mixed model, $\epsilon = 10^{-3}$.

Figure 3.4 – Zoom on Laplacian (left) and mixed (right) model adaptation for 20 Jacobi iterations.

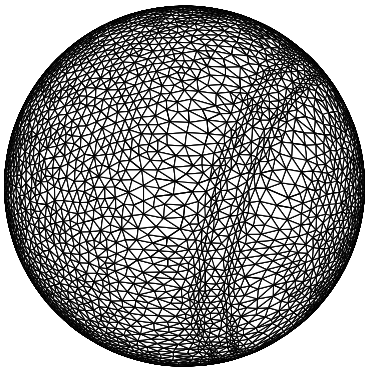
in a unit sphere on a uniform mesh with 352436 tetrahedral elements. Both the Laplacian and the mixed model are run with $\alpha = 1000, \gamma_\alpha = 1$, $\beta = 0$, $\tau = 0$ for 20 Jacobi iterations, while the mixed model employs 30 elastic iterations for each Jacobi iteration, with $\epsilon = 10^{-3}$ and $\alpha = 0$, $\beta = 0$, $\tau = 10$, $\gamma_\tau = 0.1$ for the elastic part. Results are shown in figure 3.5. The mixed model is able to reduce the element stretching caused by the Laplacian model, and the consideration presented for the two-dimensional testes still hold, with the remark that in three dimensions additional care is needed in the choice of adaptation parameters in order to avoid the possible occurrence of mesh tangling.



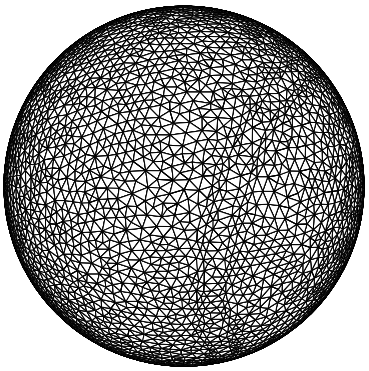
a Initial mesh and analytical function.



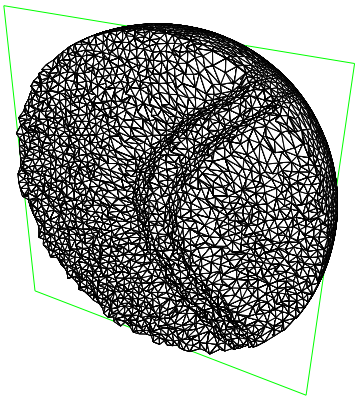
b Initial mesh and analytical function, interior cut.



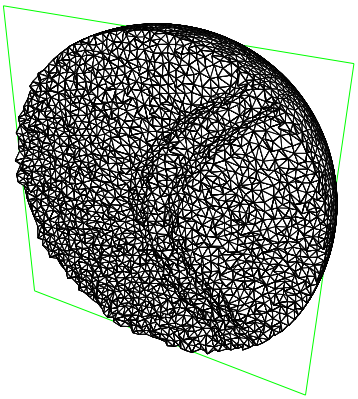
c Laplacian model adaptation.



d Mixed model adaptation.



e Laplacian model adaptation, interior cut.



f Mixed model adaptation, interior cut.

Figure 3.5 – Adaptation to an analytical function in three dimensions. Analytical function (top), Laplacian model (bottom, left), and mixed model (bottom, right).

3.9 Adaptive simulation of unsteady flows over fixed boundaries

This section is devoted to the application of dynamic mesh adaptation with constant connectivity to unsteady compressible flow simulations. The fluid flow solution is obtained through the `Flowmesh` solver described in chapter 2.

3.9.1 Two-dimensional forward facing step

A uniform Mach 3 flow of ideal gas hits a step (0.2 length unit high, located at 0.6 length units from the inlet) enclosed into a 2D wind tunnel (1 length unit wide and 3 length unit long), and a complex system of interacting nonlinear waves is developed over time. This problem has been extensively used as a benchmark to test numerical methods for compressible flows [52, 163, 176]. Slip boundary conditions are imposed on the top and bottom walls, while inlet and outlet conditions are imposed based on the characteristic directions. Although no analytical solutions are available for this problem, this case has proven to be a valuable tool for testing numerical schemes due to the difficulty in accurately reproducing several flow features as time evolves:

1. The position of reflected shocks on walls, and particularly the Mach stem on the top wall — Too large numerical dissipation causes a lag in the shock position as time evolves. Particularly, the contact discontinuity originating from the Mach stem on the top wall has to form approximately at time $t = 1.5$ in order for the shock pattern to be correct at later time instants.

2. The regular shock reflection on the bottom wall — The singularity at the step corner causes a spurious entropy layer [176], which interacts with the incoming shock, thus producing a fictious Mach reflection with a normal shock on the bottom wall too. Woodward and Colella [176] imposed an additional, artificial boundary condition on the cells surrounding the corner in order to impose constant entropy and total enthalpy throughout the time evolution, thus avoiding most of the spurious entropy production and obtaining a regular shock reflection with oblique shocks on the bottom walls. Woodward and Colella [176] and other following works (ex. [41]) which didn't impose any special treatment at the corner showed that the spurious Mach reflection converges to a regular shock reflection as the grid is refined.

The same test case has also been employed in [38]. Out initial mesh is a Delauney triangulation made of 16710 elements, 8556 nodes, with an edge length on boundary equal to 2×10^{-2}.

Results on constant-connectivity meshes are compared with anisotropic mesh adaptation with topology changes. The same density contour levels (30 equispaced lines, from the minimum to the maximum value) as in [176] are showed (figures 3.7,3.9,3.11,3.13,3.15,3.17,3.19) together with the corresponding meshes (figures 3.6,3.8,3.10,3.12,3.14,3.16,3.18).

$t = 0.5$ The main feature to be captured is the bow shock approaching the top wall. Both anisotropic and moving mesh adaptation have difficulties in catching the tail of the shock, the former being worse due to the local mesh derefinement. Anisotropic mesh adaptation manages to adapt on part of the expansion fan stemming from the step corner, while gradient-based moving mesh adaptation mostly adapts on the moving shock.

$t = 1$ The shock has impacted on the top wall, and the reflected shock is interacting with the corner expansion fan. Although the moving mesh doesn't adapt on the expansion fan, the shock appears to be correctly curved by the interaction.

$t = 1.5$ The reflected shock now impacts the bottom wall. This reflection is a regular one, with oblique shocks, while the reflection on the top wall is a Mach reflection, with a normal shock connecting the incoming and outcoming shocks plus a contact discontinuity originating from the intersection. The stem and contact discontinuity begin to be visible at this time, and the anisotropic mesh adaptation gives the best results on the resolution of the forming contact discontinuity.

$t = 2$ The wave pattern is moving downstream, curving under the effects of the interaction with the corner expansion fan. Results on the three simulations are mostly comparable.

$t = \{2.5, 3, 4\}$ As before, but now the effects of the interaction of the regular shock reflection with the spurious entropy layer on the top wall begin to be visible. The anisotropic mesh adaptation gives the best results in preserving the regular shock reflection.

The Laplacian model is run using mass density as the scalar solution to drive the adaptation, with parameter values $\alpha = 40$, $\gamma_\alpha = 0.1$, $\beta = 10$, $\gamma_\beta = 0.5$, $\tau = 0$ with 10 Jacobi iterations. The mixed model, instead, is run using the Mach number as the scalar solution to drive the adaptation, with parameter values $\alpha = 40$, $\gamma_\alpha = 0.05$, $\beta = 10$, $\gamma_\beta = 0.5$, $\tau = 0$ for the Laplacian part, and parameter values $\alpha = 10$, $\gamma_\alpha = 0.5$, $\beta = 10$, $\gamma_\beta = 0.05$, $\tau = 0$ for the elastic part, with $\epsilon = 5 \times 10^{-4}$. The difference is due to the fact that the mixed model reduces the element stretching produced by the Laplacian iterations, so it is possible to use more aggressive

	Anisotropic	Laplacian model	Mixed model
$N°$ elements at $t = 4$	10801	16710	16710
$N°$ nodes at $t = 4$	5511	8556	8556
Computational time	49m 18s	1h 16m 54 s	57m 31s

Table 3.1 – Mesh statistics and computational times for the forward facing step cases.

parameters for the Laplacian part, thus recovering some adaptation effect on the expansion fan starting from the corner, and on the contact surface starting from the triple point.

Table 3.1 compares mesh nodes and computational time for anisotropic adaptation with topology change, Laplacian model and mixed model. All simulations are run on 4 CPUs, mesh adaptation is serial. Anisotropic adaptation provides the most competitive computational times, but this can be due to the significant reduction in the number of nodes. It is interesting to note that the mixed model, notwithstanding the increase in computational operations, manages to be faster than the Laplacian model. This is due to the reduced element stretching in the direction orthogonal to solution fronts, which ease the convergence of the flow solver thus reducing the number of iterations in the dual time step integration.

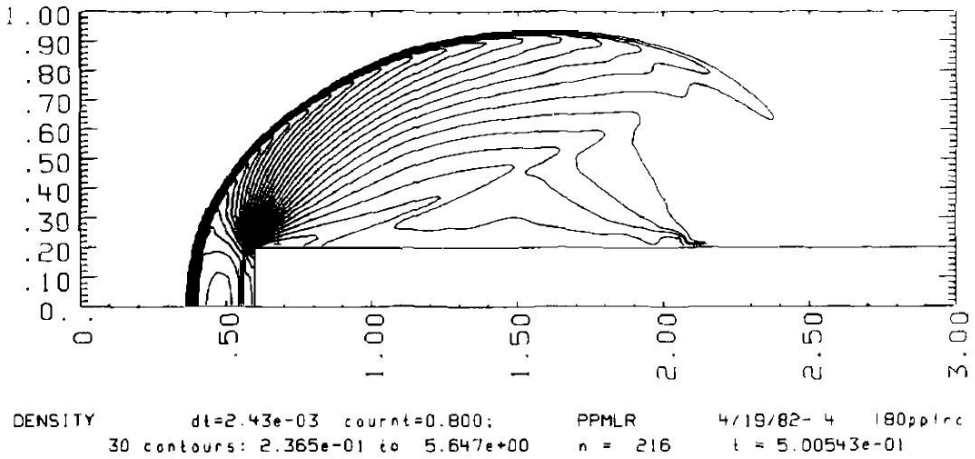
3.9.2 Three-dimensional forward facing step

A preliminary application to a three-dimensional unsteady simulation is done through a three-dimensional extrusion of the forward facing step geometry. The unsteady flow retains its two-dimensional nature, but specific issues related to three-dimensionality appears in mesh adaptation, as the mesh needs to move on edges and corners of the domain, with corresponding algorithmic requirements. Results for the Laplacian and mixed models are shown in figures 3.20, 3.21, 3.22.

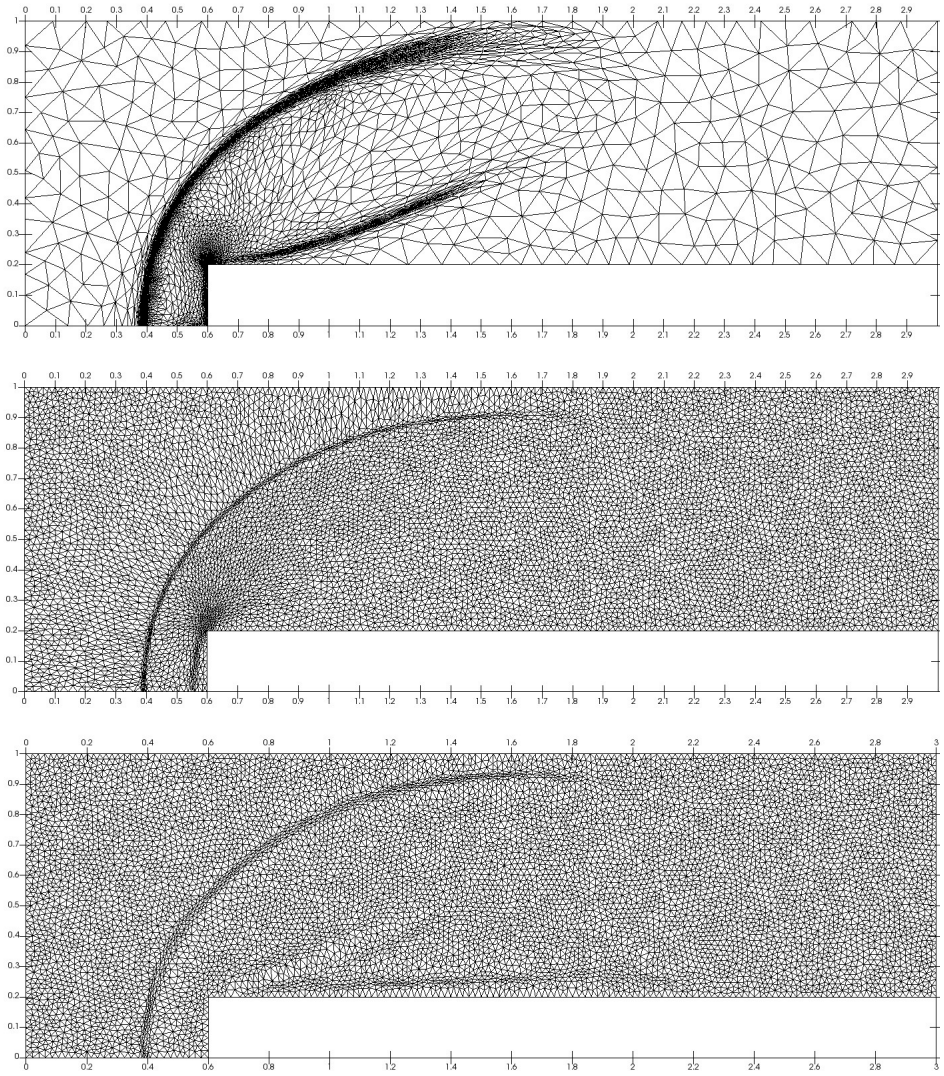
In three dimensions, the element stretching produced by the Laplacian model is even more pronounced.

3.10 Adaptive simulation of unsteady flows over moving boundaries in 2D

In this section, the dynamic mesh adaptation strategy over moving boundaries presented in section 3.6 is applied to an aeronautical flow. The analysis is restricted to the Laplacian model, as with moving boundaries the effect of algebraic system iterations does not accumulate across time steps, so the difference

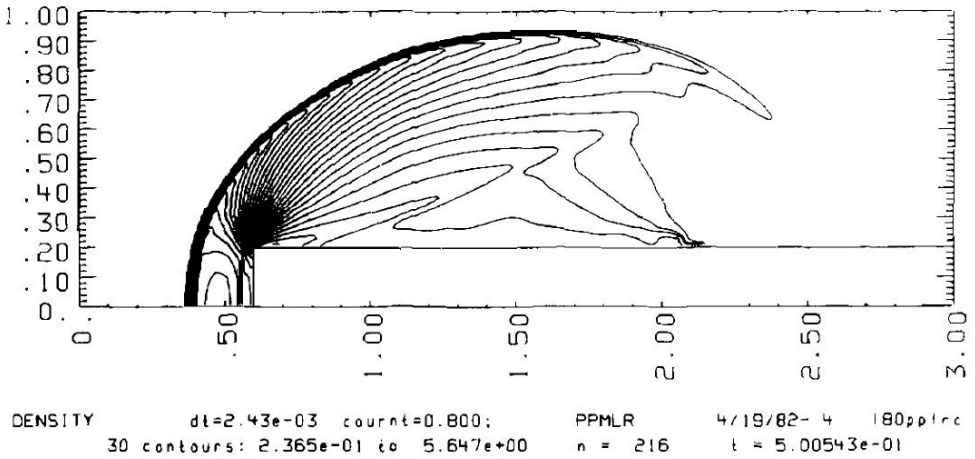


a Density contours from [176].

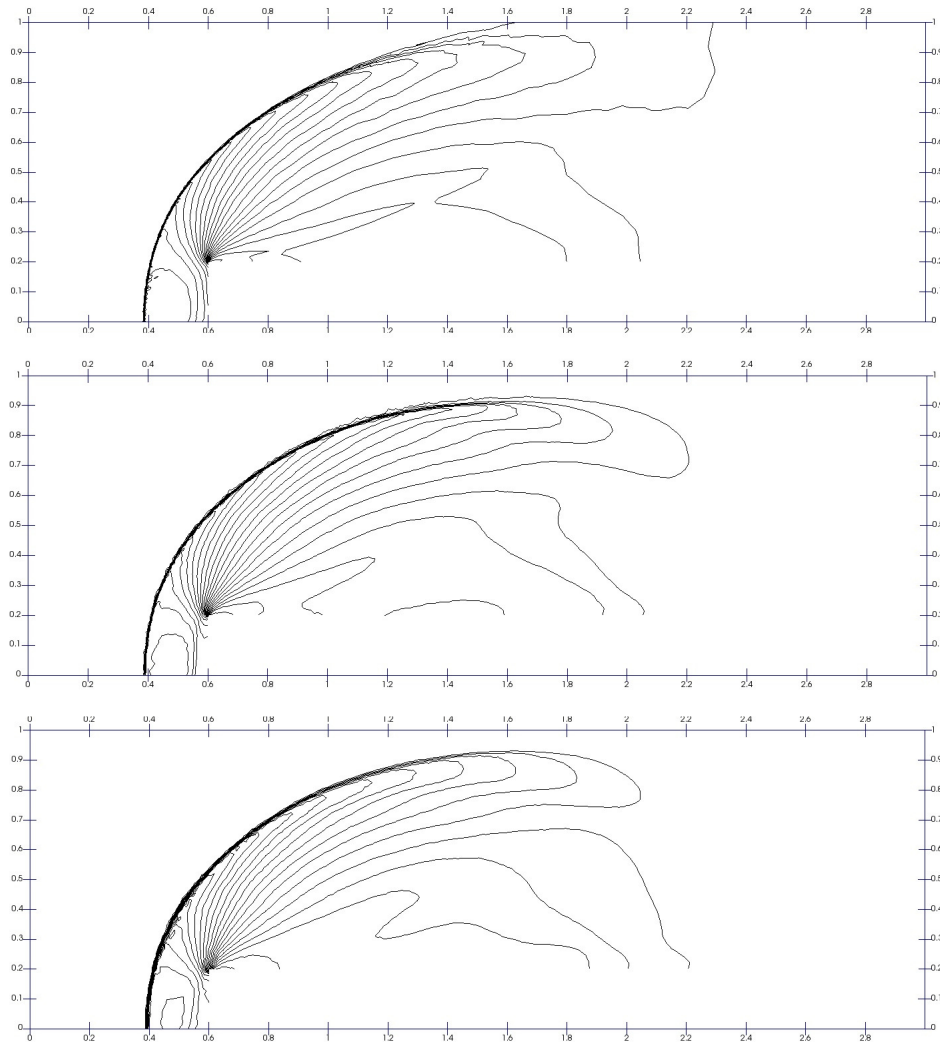


b Meshes with anisotropic adaptation, Laplacian model and mixed model.

Figure 3.6 – Forward facing step meshes at $t = 0.5$.

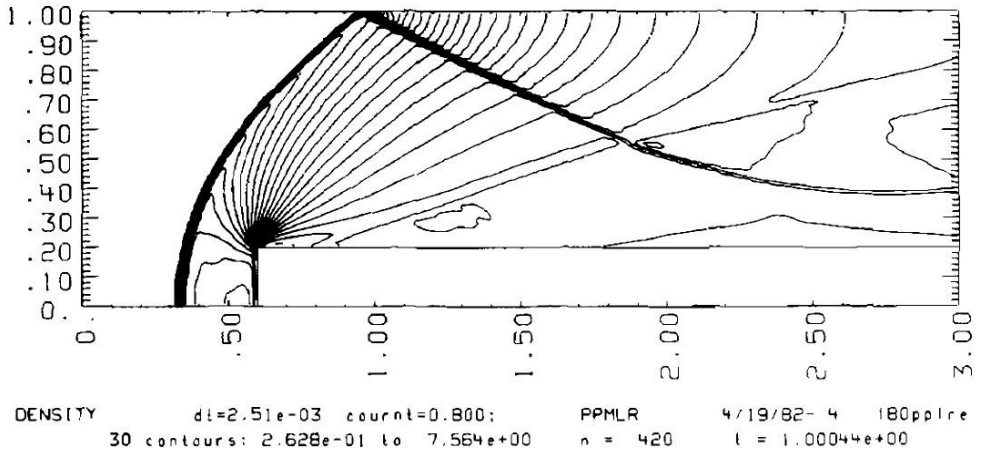


a Density contours from [176].

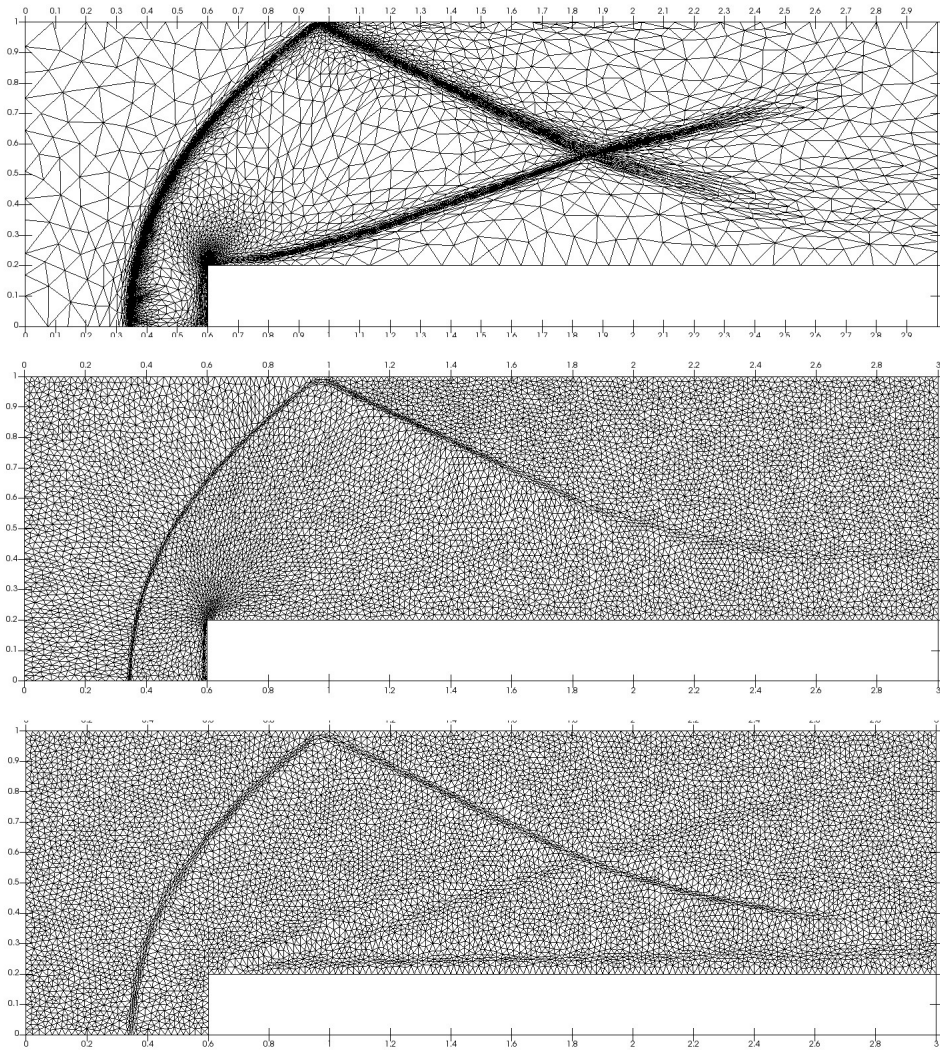


b Density contours with anisotropic adaptation, Laplacian model and mixed model.

Figure 3.7 – Forward facing step results at $t = 0.5$.

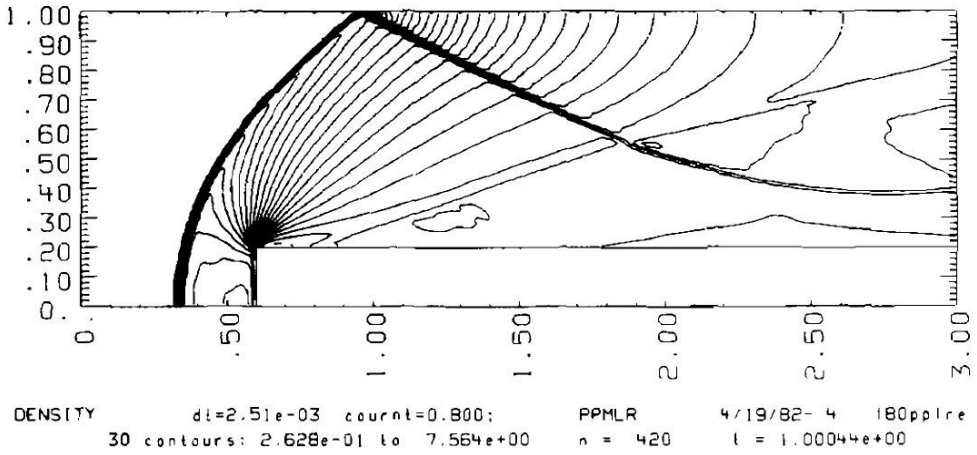


a Density contours from [176].

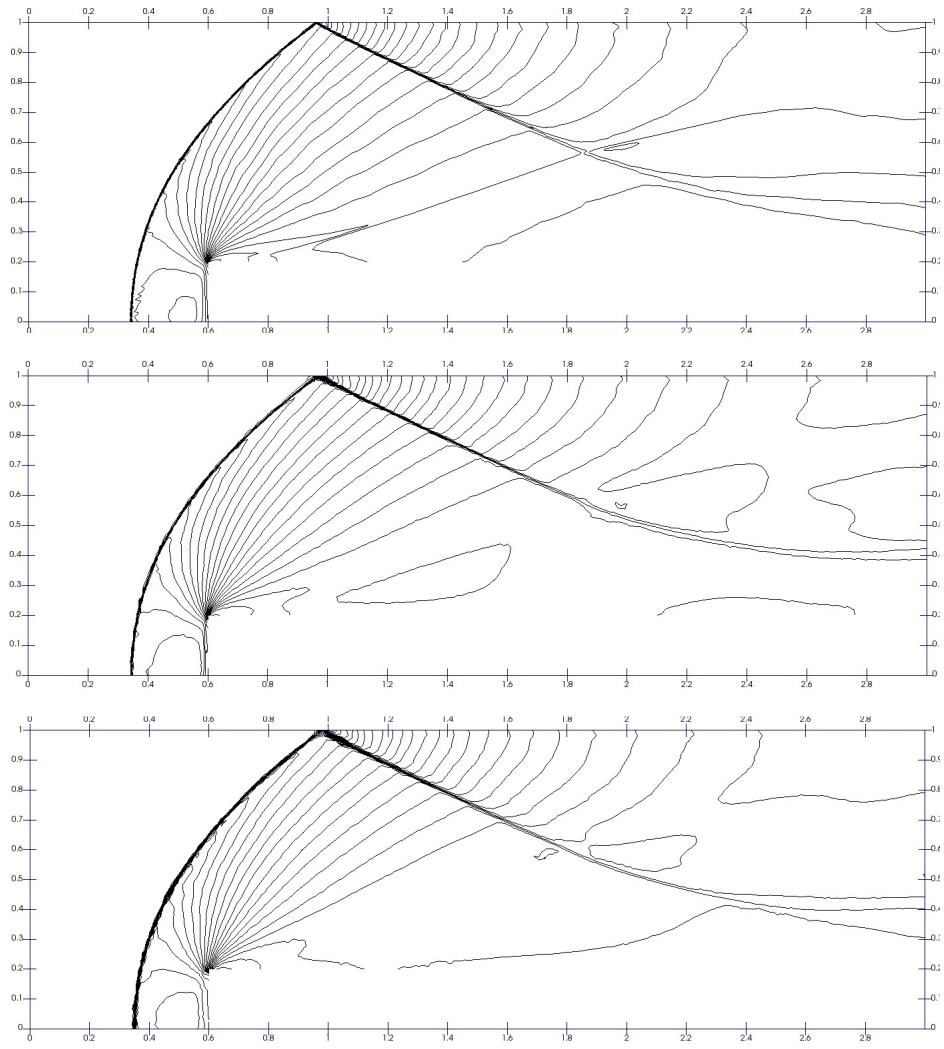


b Meshes with anisotropic adaptation, Laplacian model and mixed model.

Figure 3.8 – Forward facing step meshes at $t = 1.0$.

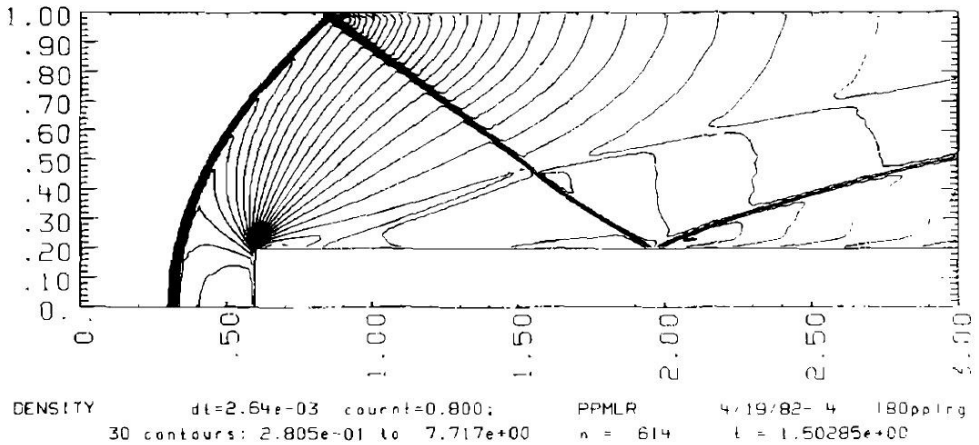


a Density contours from [176].

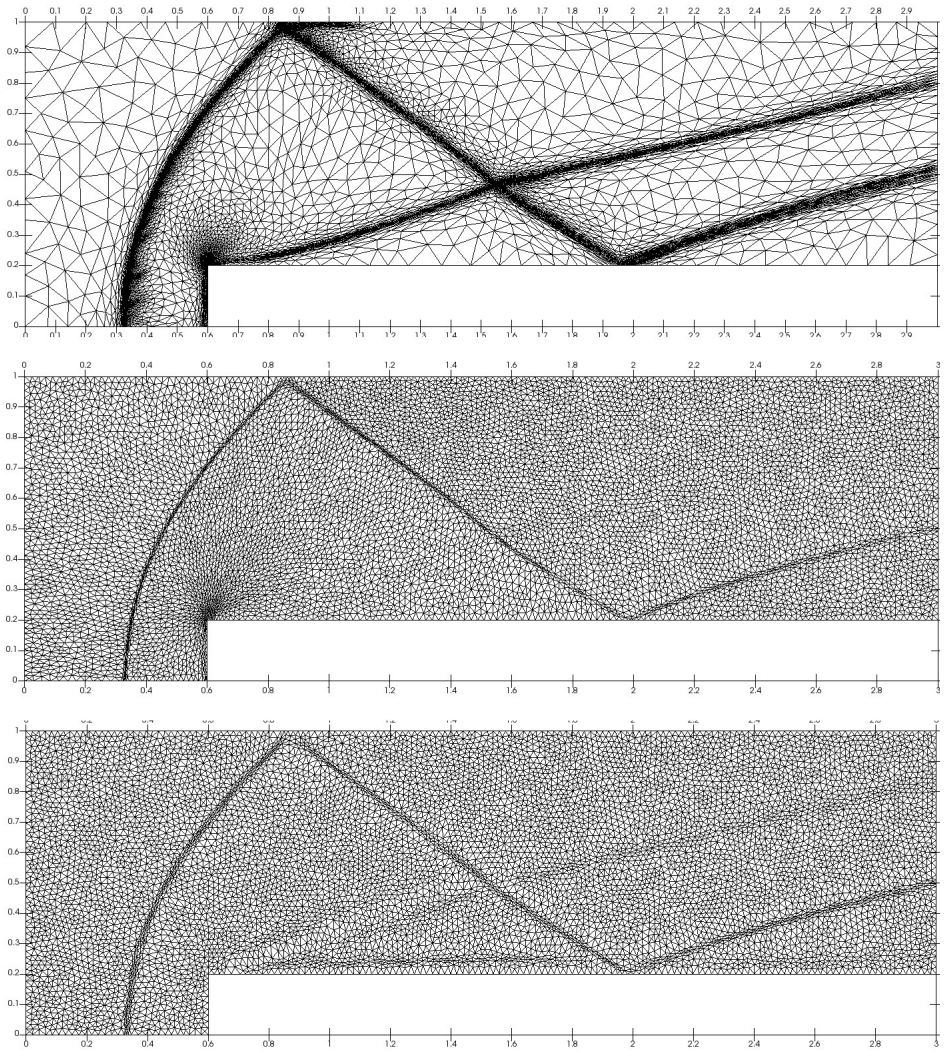


b Density contours with anisotropic adaptation, Laplacian model and mixed model.

Figure 3.9 – Forward facing step results at $t = 1.0$.

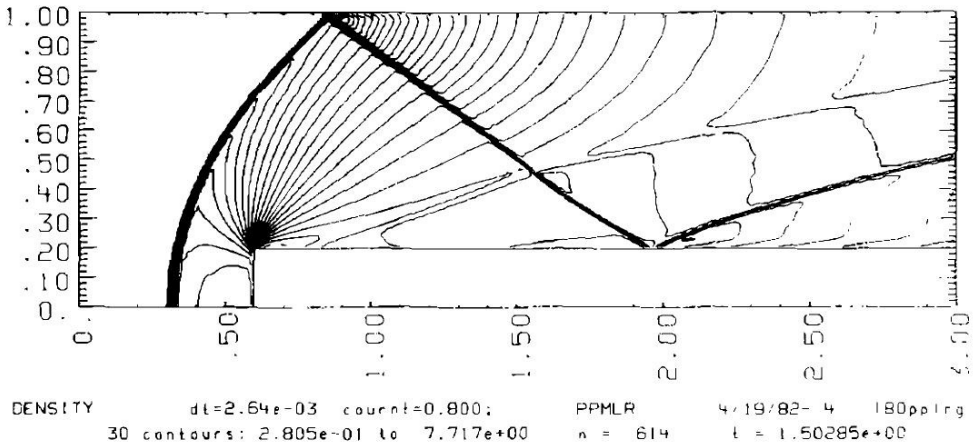


a Density contours from [176].

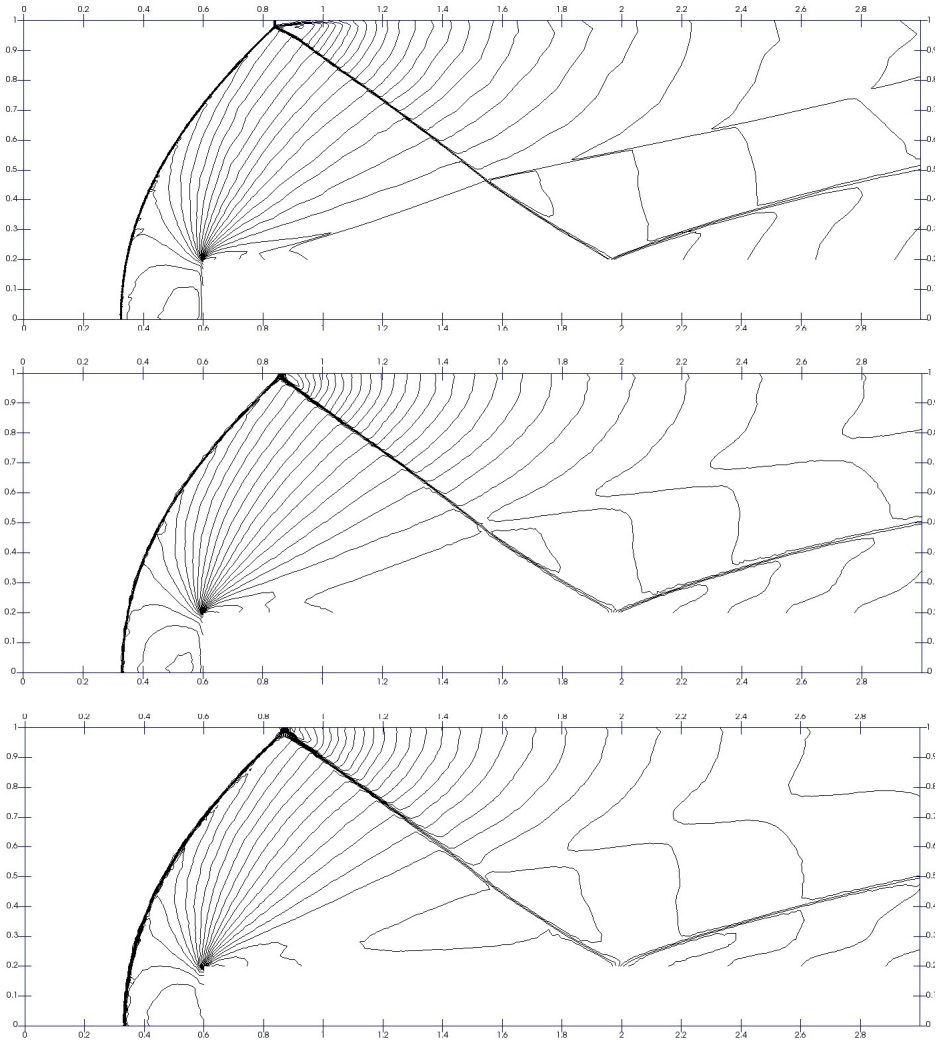


b Meshes with anisotropic adaptation, Laplacian model and mixed model.

Figure 3.10 – Forward facing step meshes at $t = 1.5$.

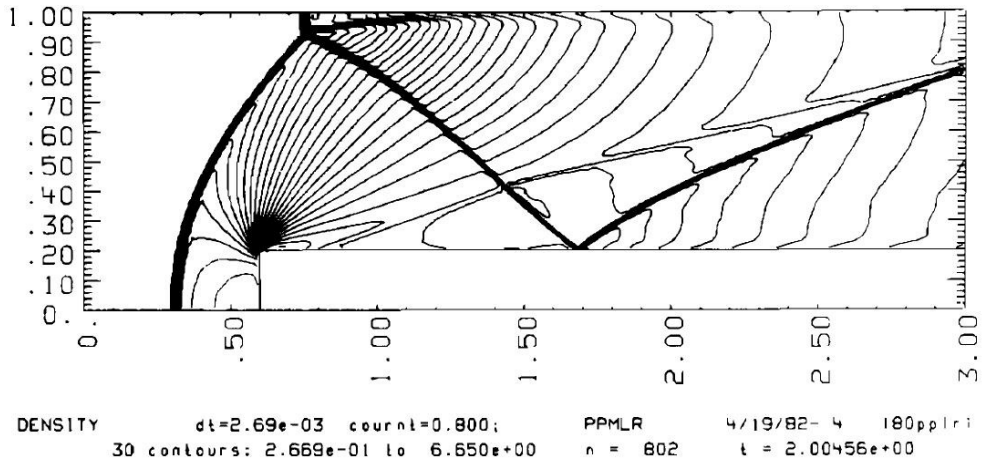


a Density contours from [176].

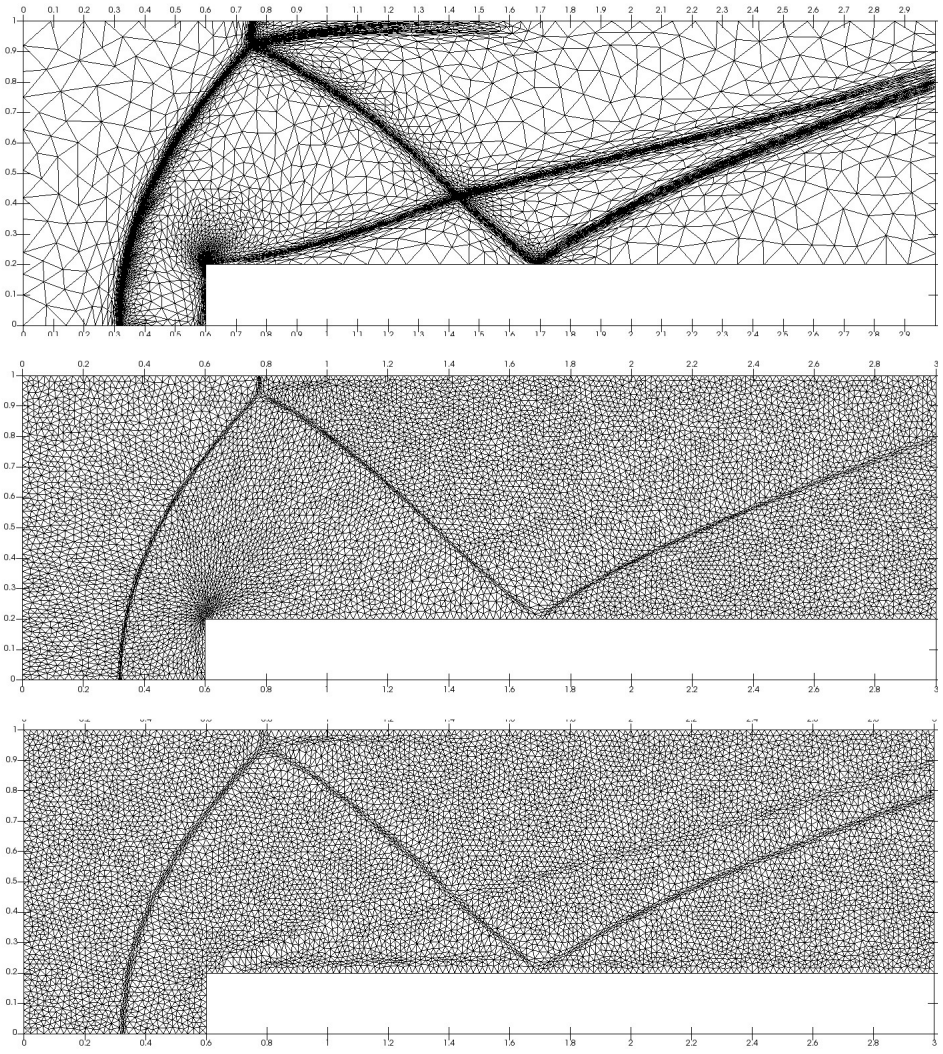


b Density contours with anisotropic adaptation, Laplacian model and mixed model.

Figure 3.11 – Forward facing step results at $t = 1.5$.

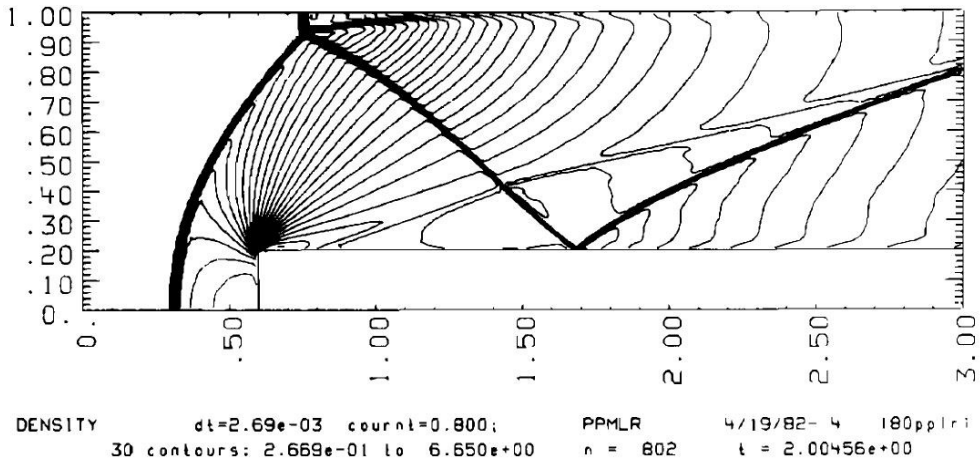


a Density contours from [176].

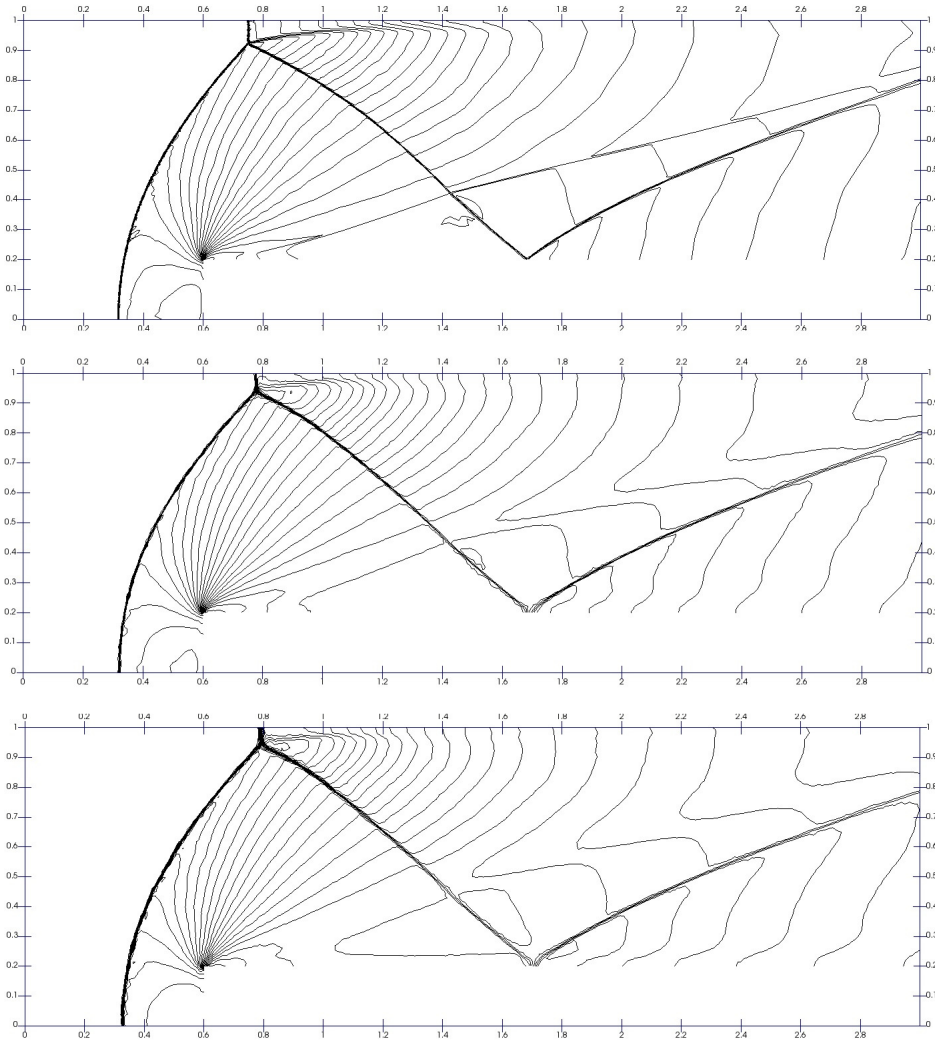


b Meshes with anisotropic adaptation, Laplacian model and mixed model.

Figure 3.12 – Forward facing step meshes at $t = 2.0$.

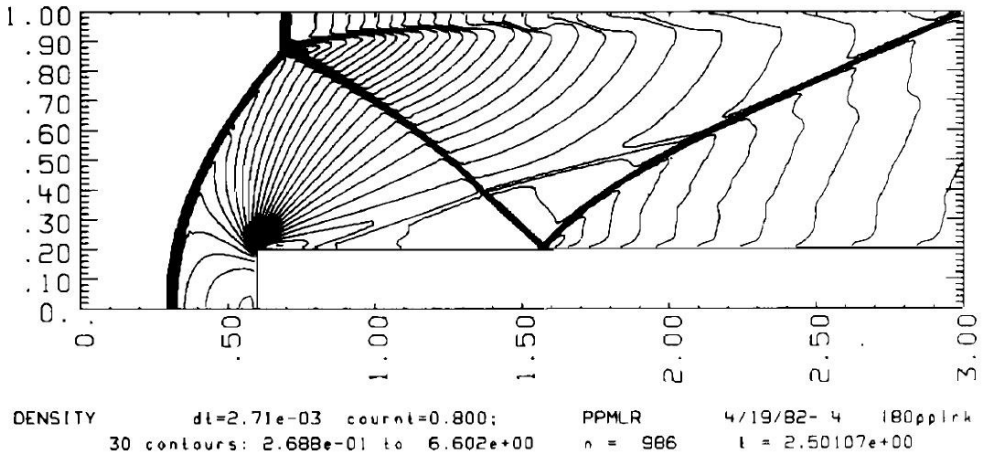


a Density contours from [176].

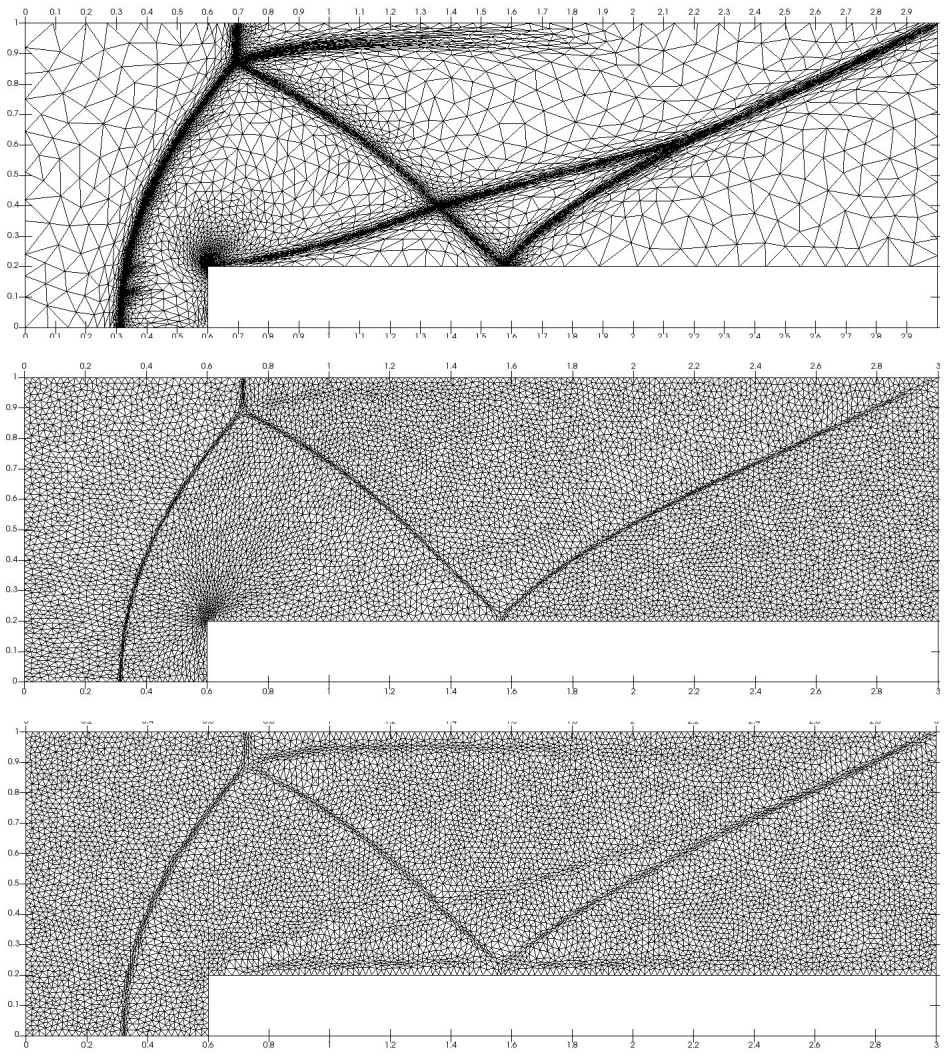


b Density contours with anisotropic adaptation, Laplacian model and mixed model.

Figure 3.13 – Forward facing step results at $t = 2.0$.

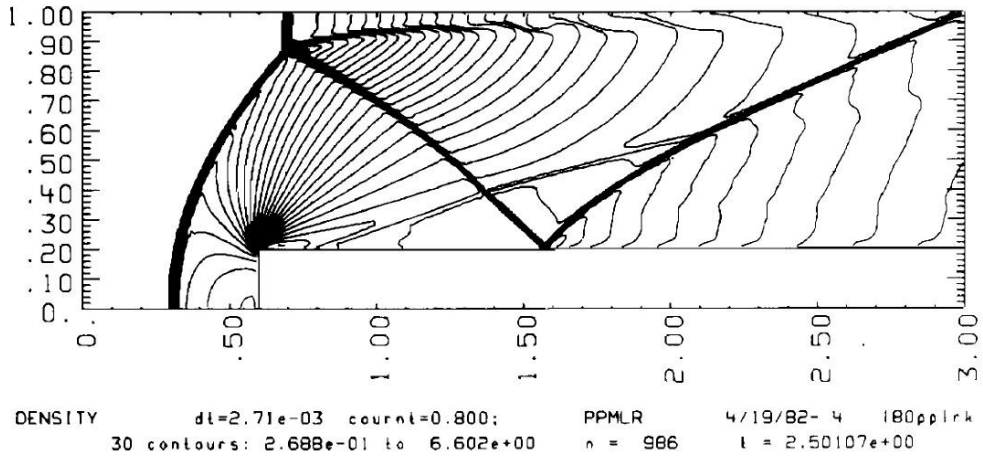


a Density contours from [176].

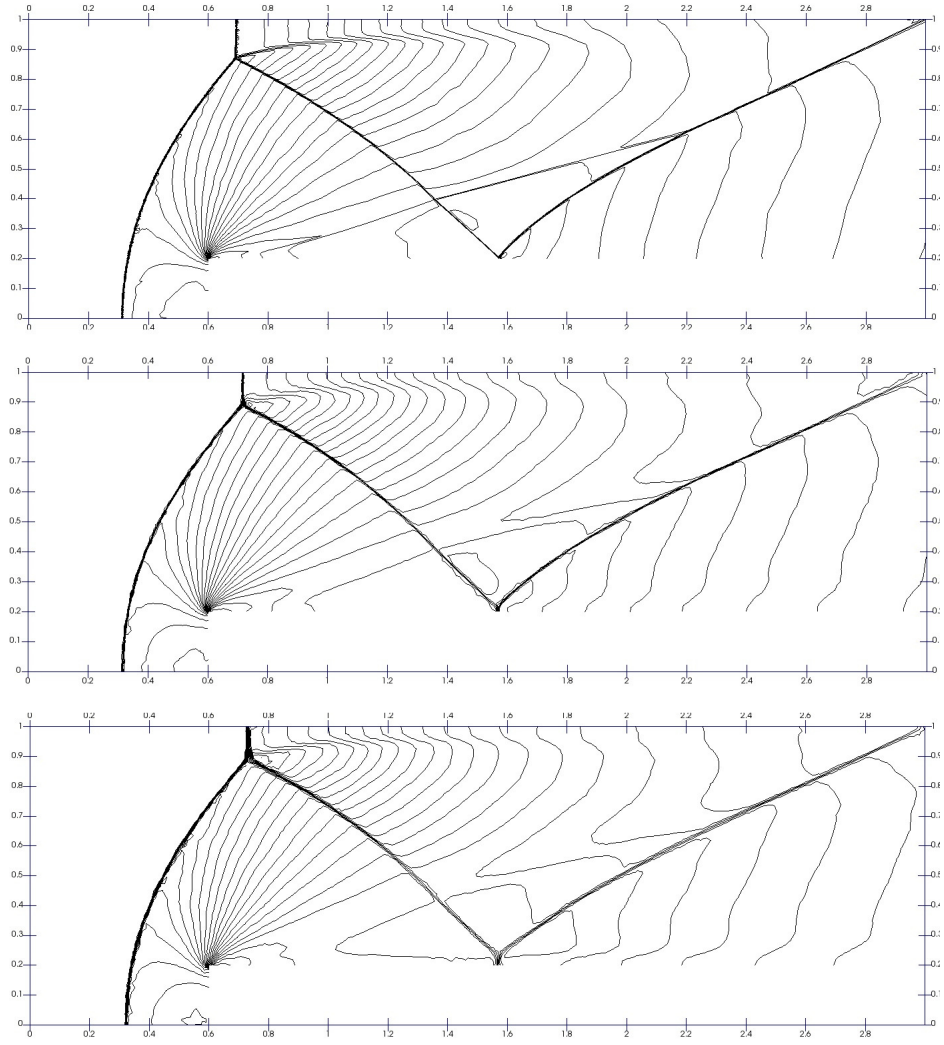


b Meshes with anisotropic adaptation, Laplacian model and mixed model.

Figure 3.14 – Forward facing step meshes at $t = 2.5$.

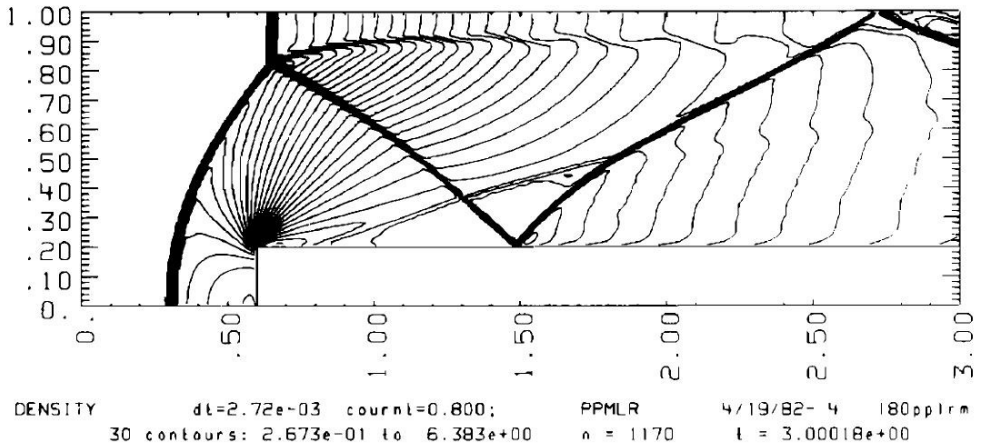


a Density contours from [176].

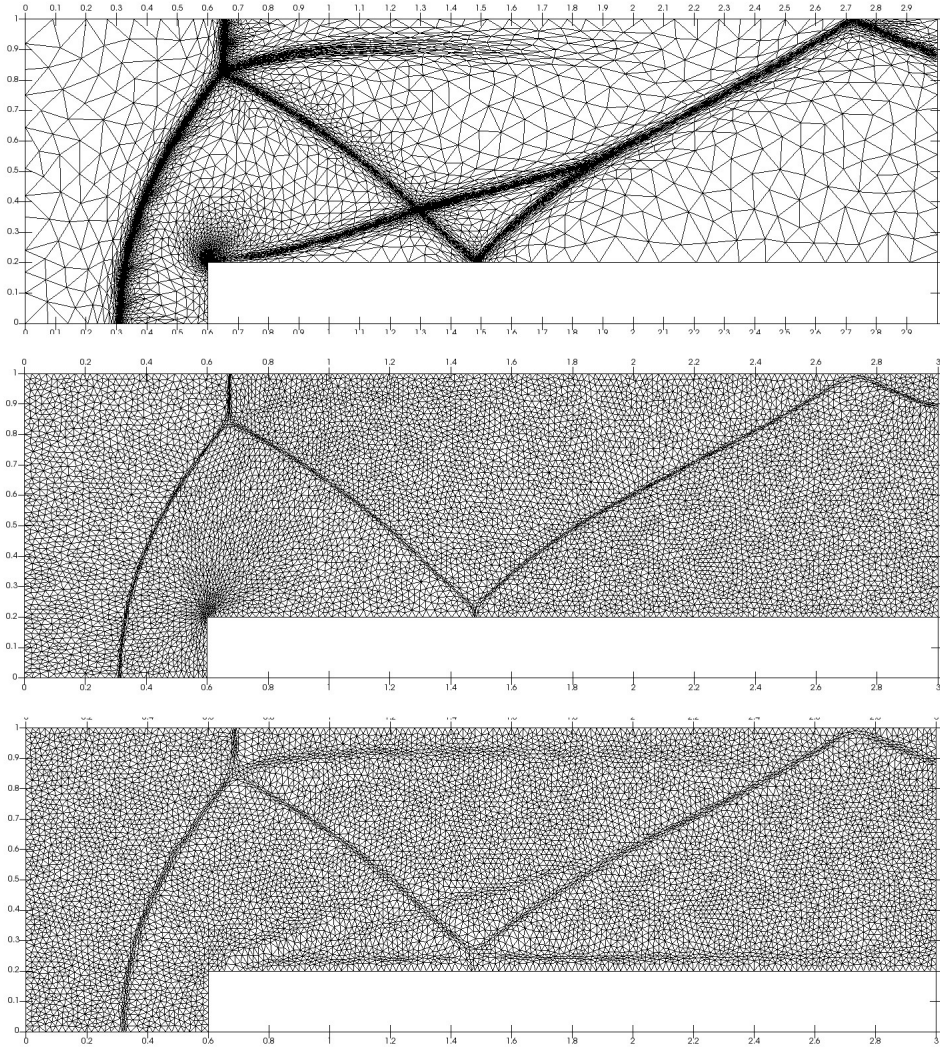


b Density contours with anisotropic adaptation, Laplacian model and mixed model.

Figure 3.15 – Forward facing step results at $t = 2.5$.

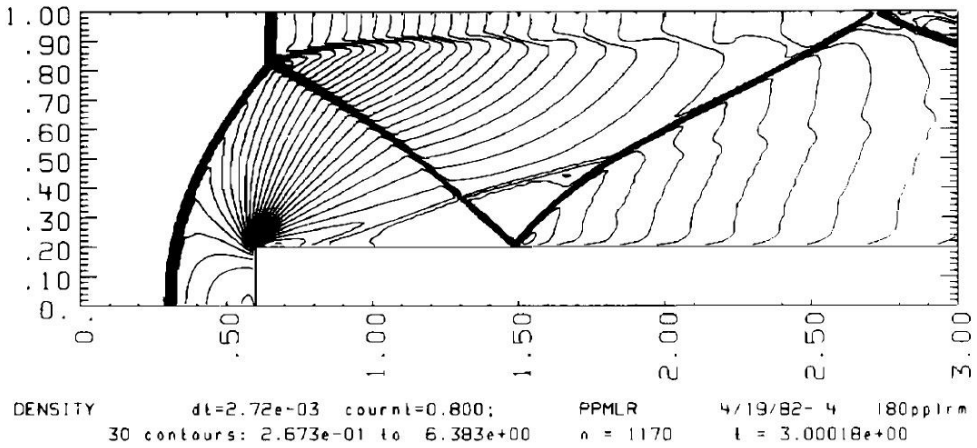


a Density contours from [176].

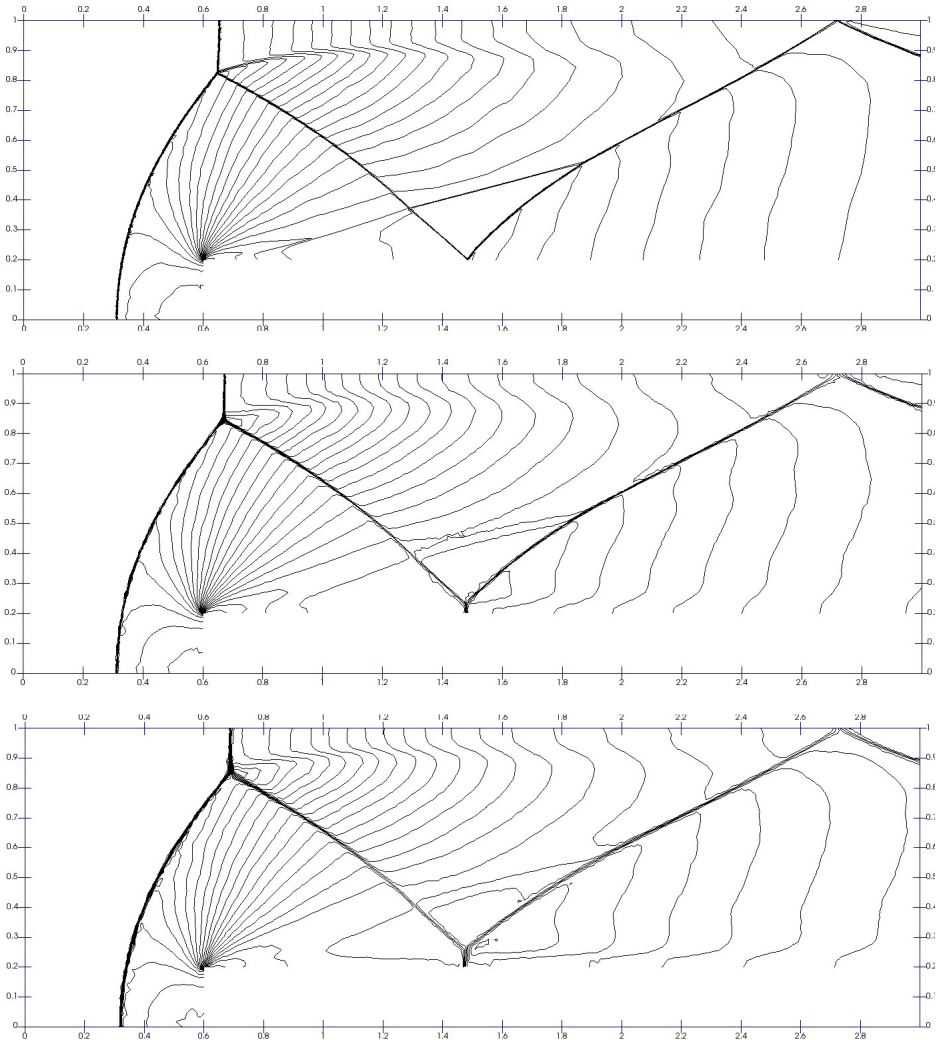


b Meshes with anisotropic adaptation, Laplacian model and mixed model.

Figure 3.16 – Forward facing step meshes at $t = 3.0$.

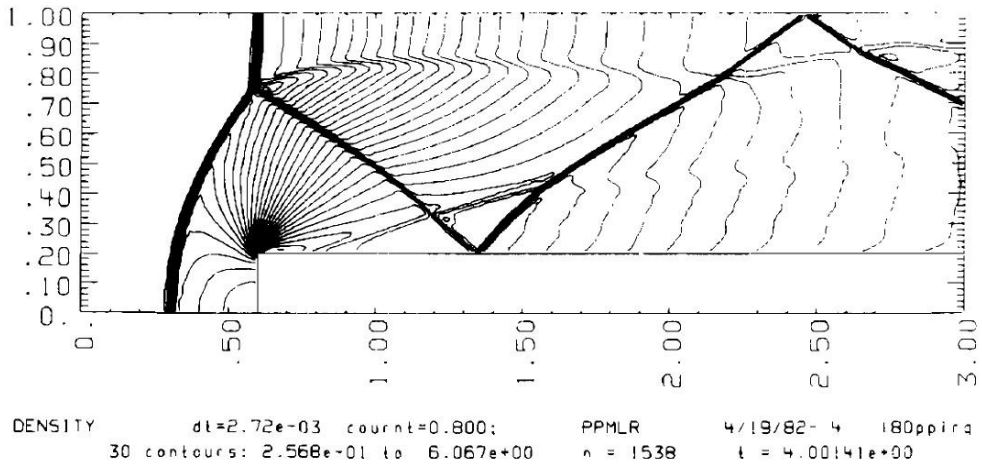


a Density contours from [176].

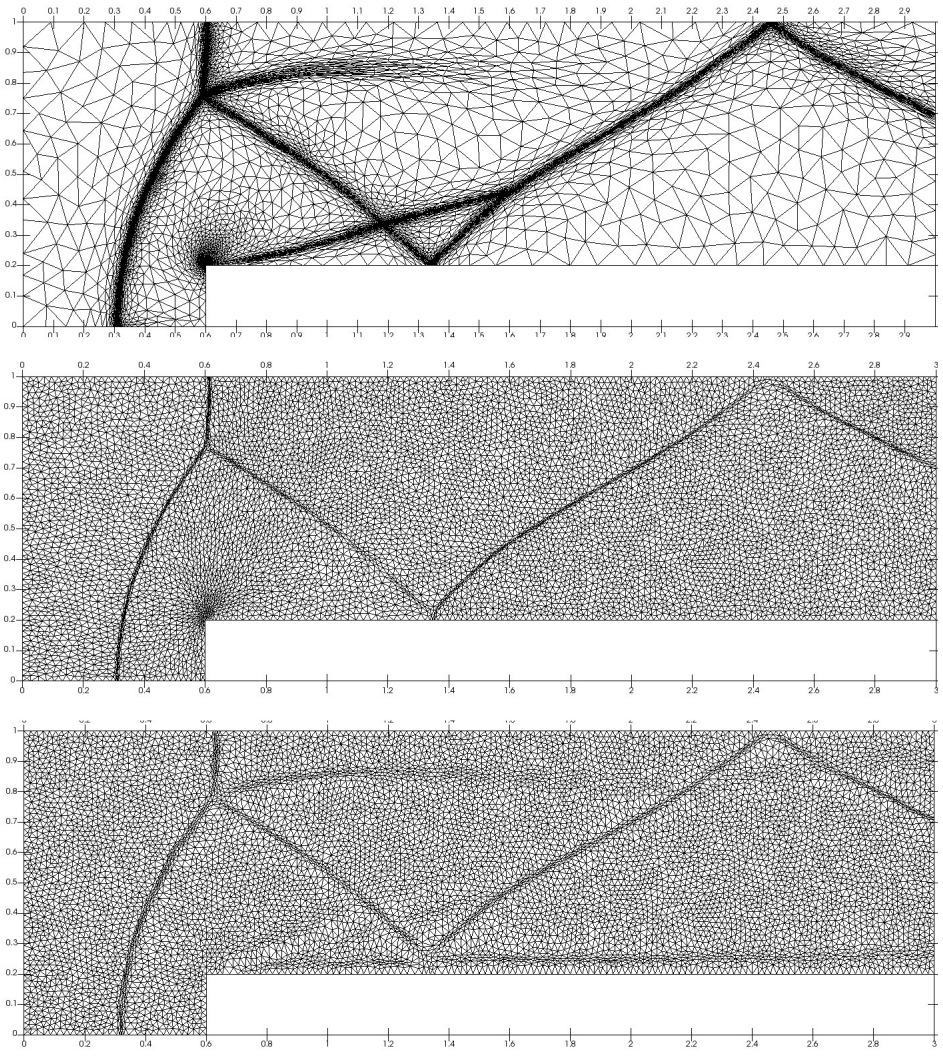


b Density contours with anisotropic adaptation, Laplacian model and mixed model.

Figure 3.17 – Forward facing step results at $t = 3.0$.

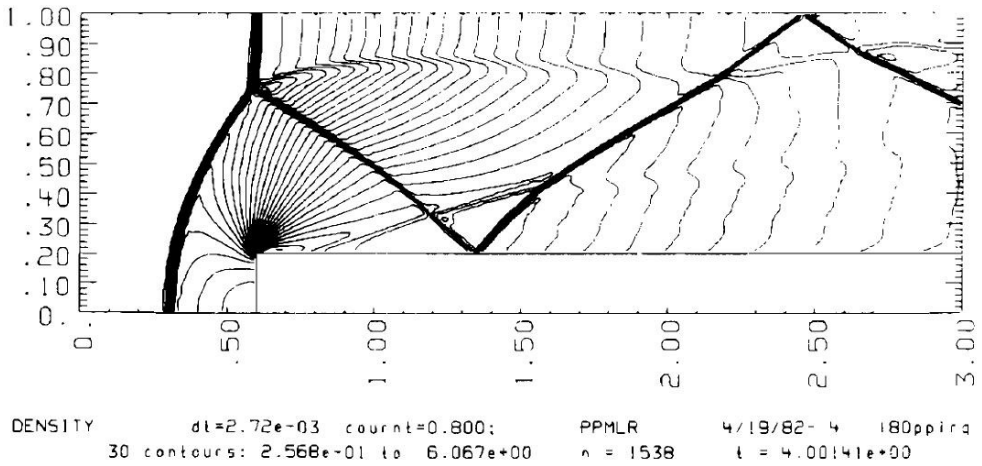


a Density contours from [176].

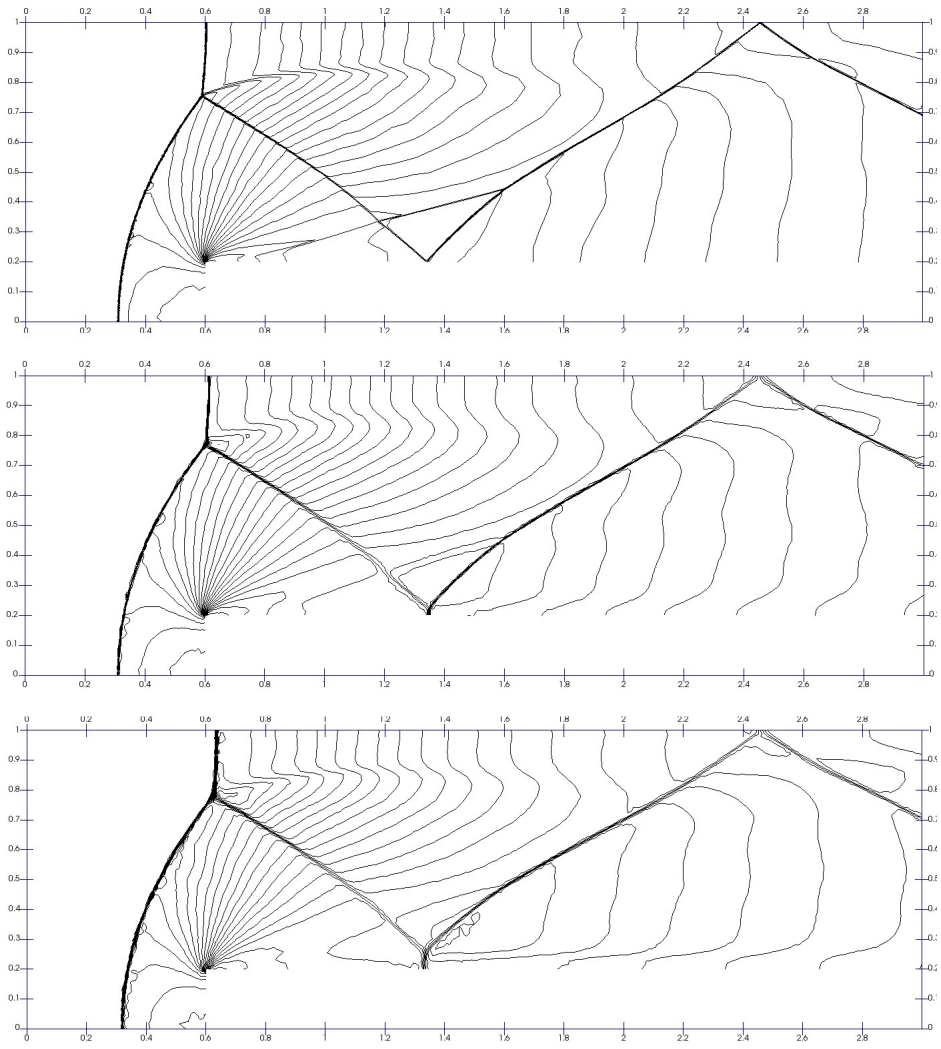


b Meshes with anisotropic adaptation, Laplacian model and mixed model.

Figure 3.18 – Forward facing step meshes at $t = 4.0$.

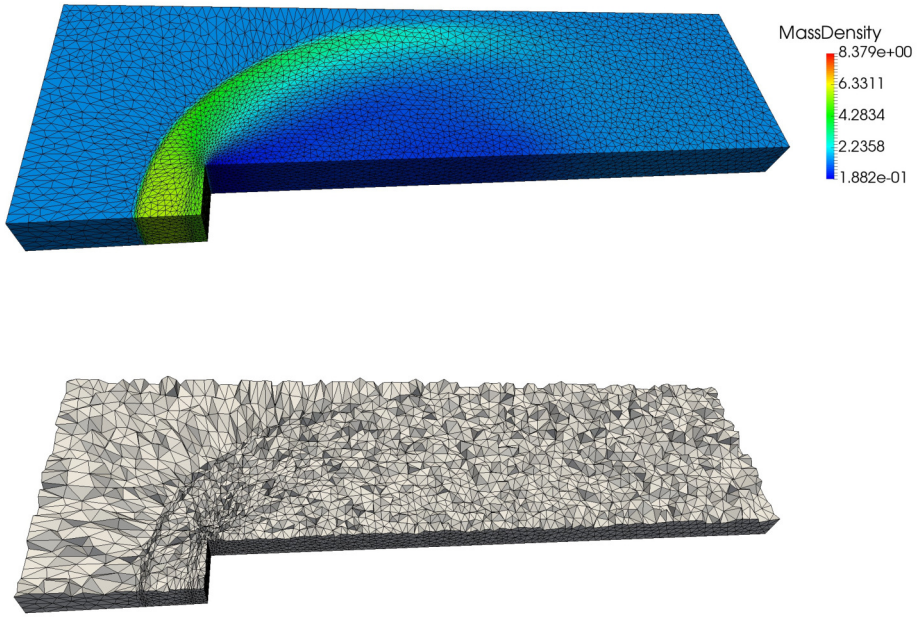


a Density contours from [176].

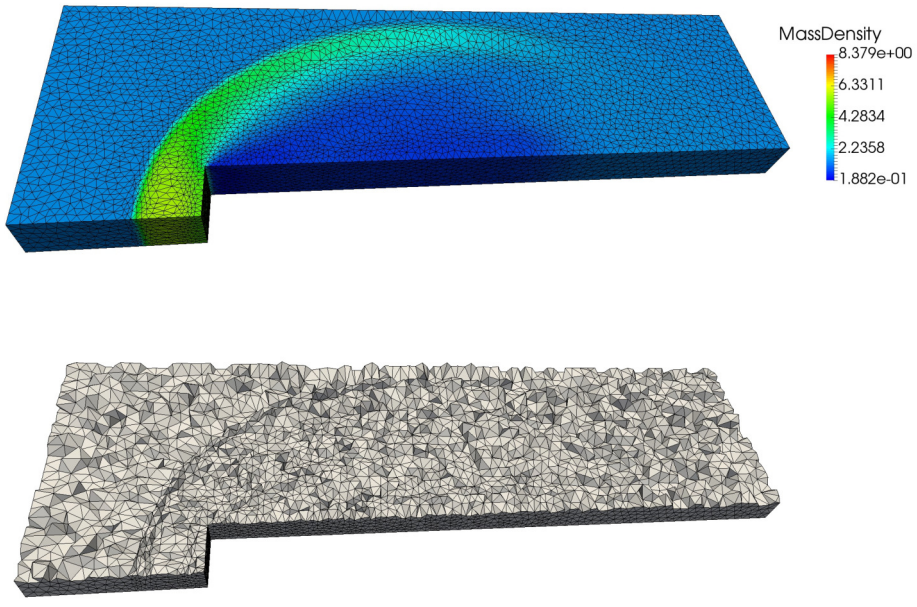


b Density contours with anisotropic adaptation, Laplacian model and mixed model.

Figure 3.19 – Forward facing step results at $t = 4.0$.

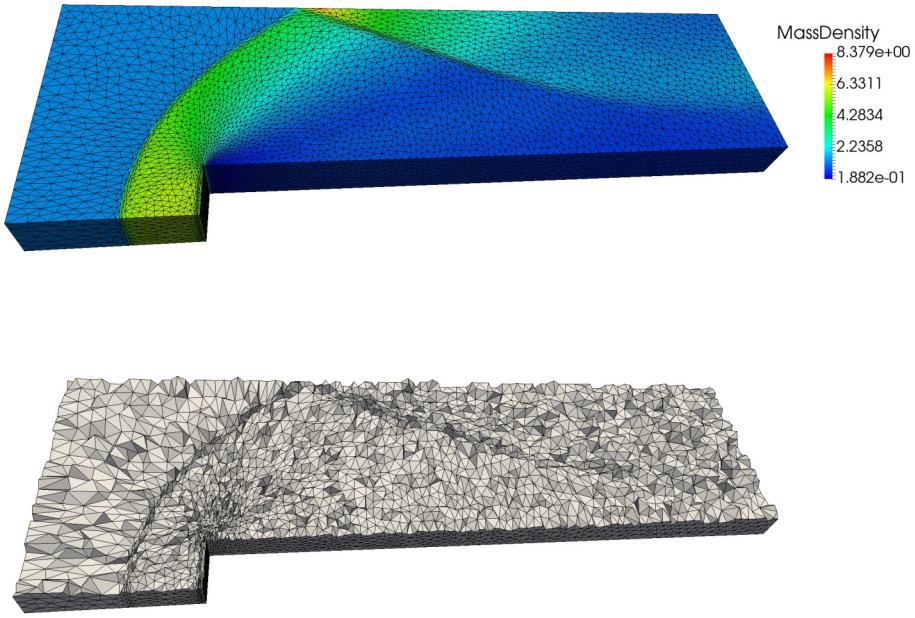


a Laplacian model, boundary and volumic cut.

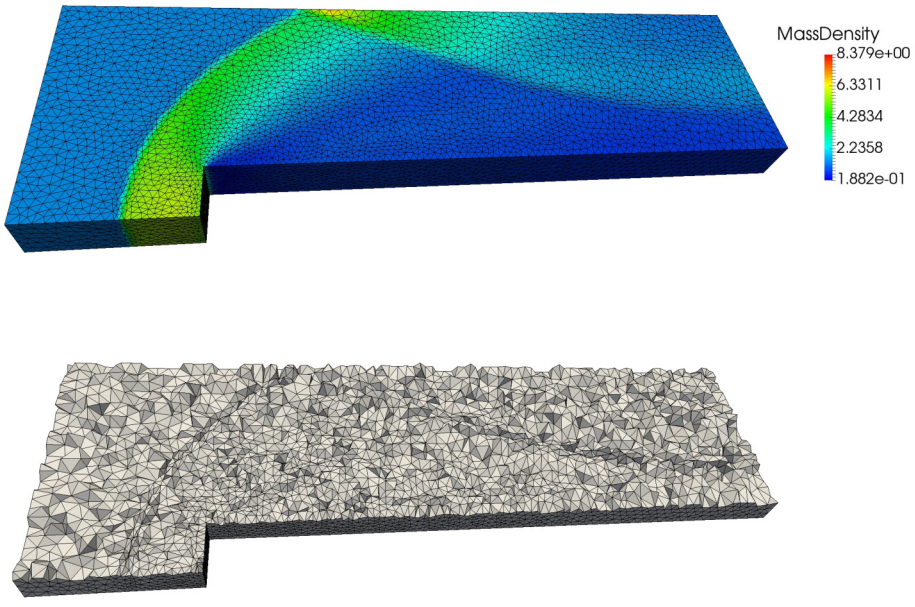


b Mixed model, boundary and volumic cut.

Figure 3.20 – Results at $t = 0.5$.

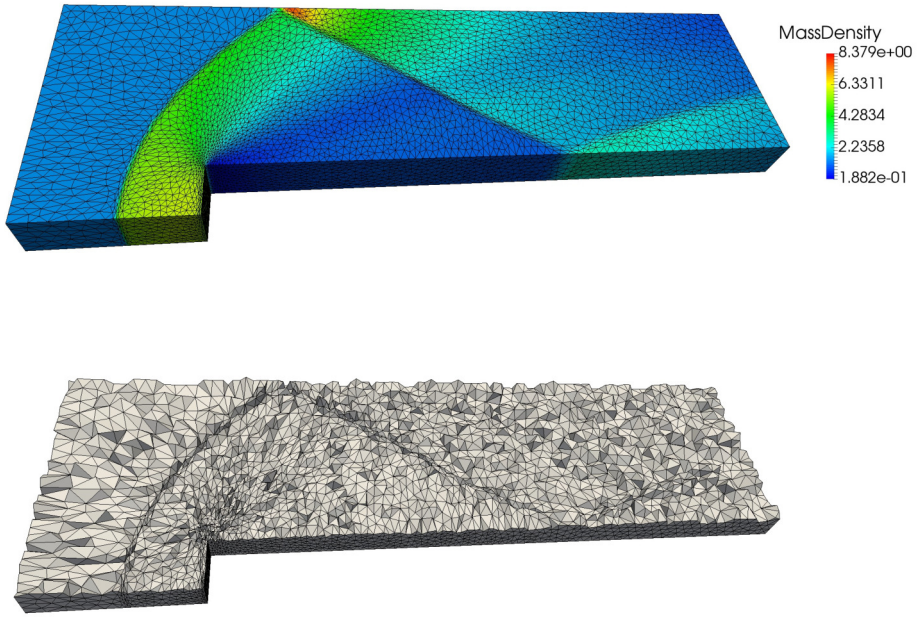


a Laplacian model, boundary and volumic cut.

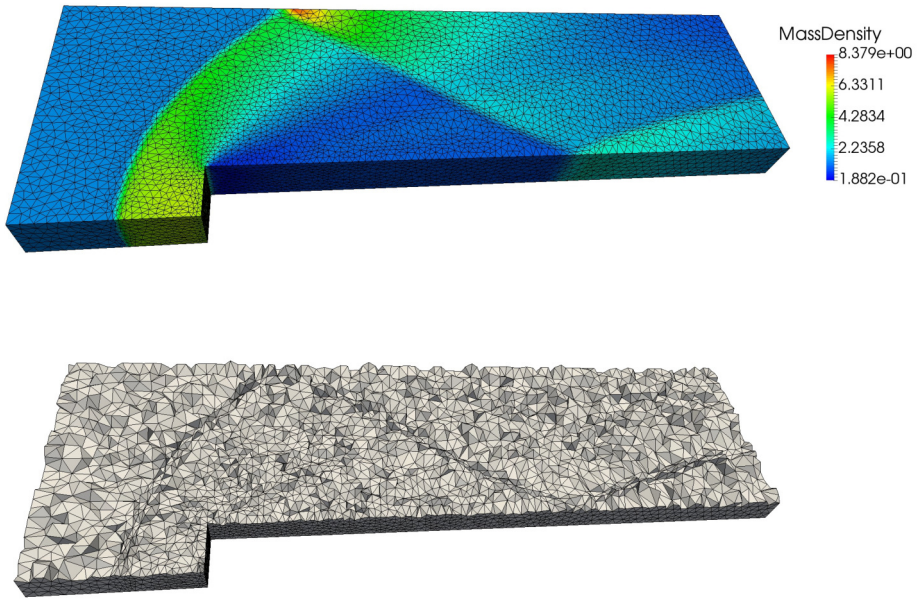


b Mixed model, boundary and volumic cut.

Figure 3.21 – Results at $t = 1.0$.



a Laplacian model, boundary and volumic cut.



b Mixed model, boundary and volumic cut.

Figure 3.22 – Results at $t = 1.5$.

between Laplacian and mixed model remains feeble for few iterations, like is the case in coupling the adaptation procedure with an unsteady flow solver.

3.10.1 Pitching NACA 0012 airfoil

We consider a NACA0012 airfoil pitching about an axis located at 0.25 its chord with a time law

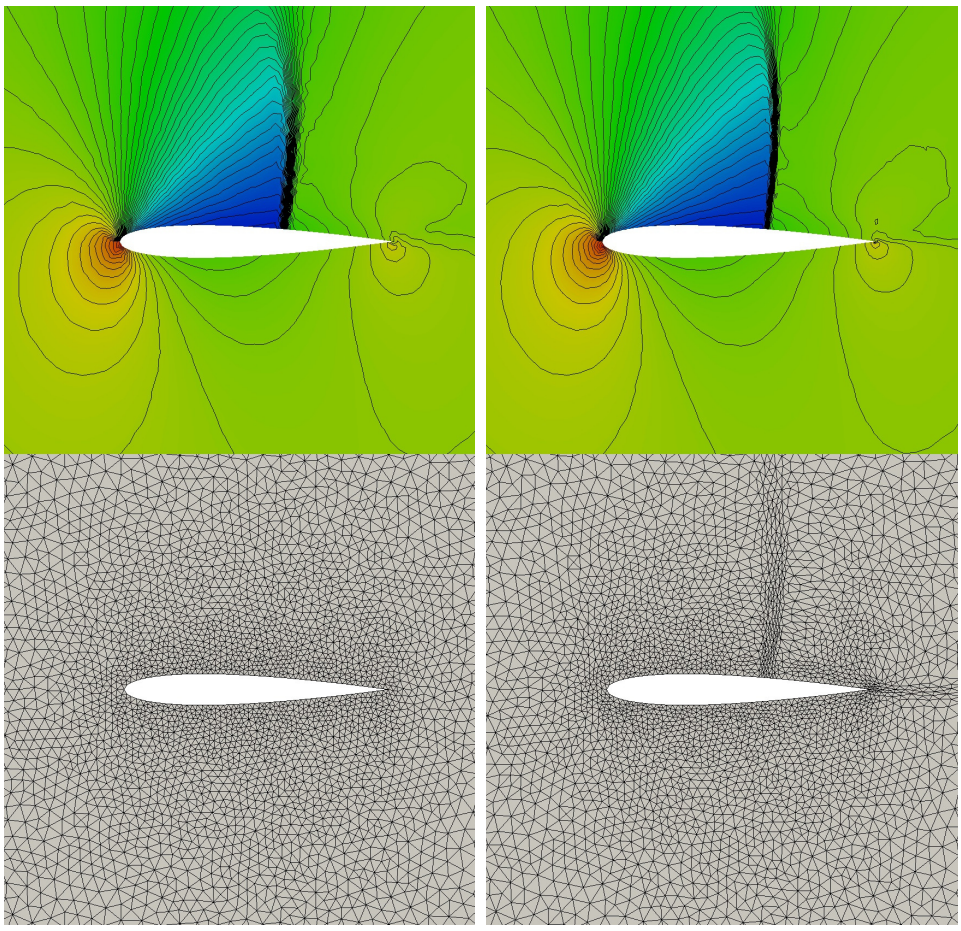
$$\alpha(t) = \Delta\alpha \cos(\kappa t) \tag{3.52}$$

with $\kappa = 0.1628$, $\Delta\alpha = 5°$ in a $M = 0.755$ flow. The transonic flow exhibits a shock wave alternatively appearing and disappearing on the upper and lower surface. This shock wave can be considered as a "type B" shock according to the classification given by Tijdeman [158], since it completely disappears from one surface as it reappears on the other one.

We test the capability of the r-adaptation technique to follow the shock wave motion. Figures 3.23, 3.24 show the application of the Laplacian model on a triangular mesh with 17412 elements, with an edge length of 0.02 time the chord on the airfoil. Laplacian parameter values are $\alpha = 1000$, $\gamma_\alpha = 0.1$, $\beta = 100$, $\gamma_\beta = 0.5$, $\tau = 0$ with 20 Jacobi iterations.

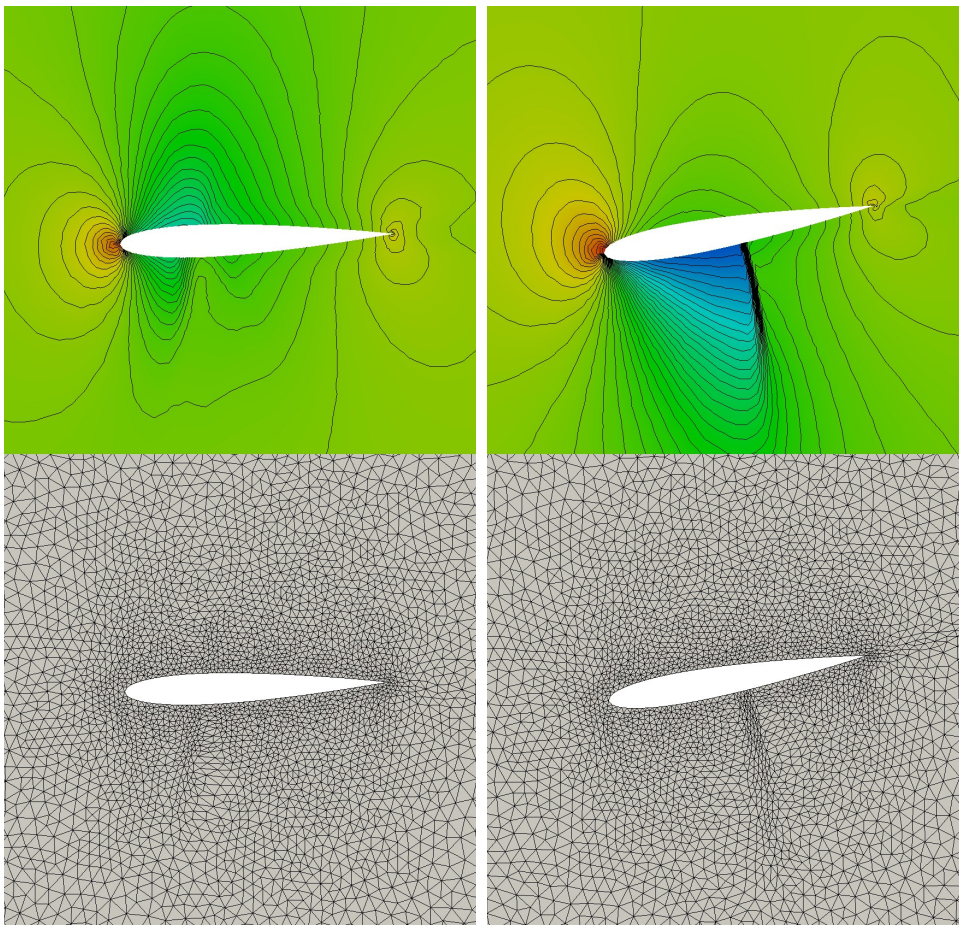
It is interesting to notice that, during the transition of the shock wave motion between upper and lower surface, the adaptation seams to follow some less intense waves appearing right aft of the disappearing shock wave. To investigate whether this is a numerical artifact, or not, we have performed the same simulation on a refined mesh with 57652 elements. As shown in figure 3.25, this flow features are confirmed with mesh refinement, and are consistent with the wavelets anecdotically reported by Tijdeman [158] as less intense waves appearing as a shock wave disappears during an oscillation cycle of an airfoil or flap (an original picture is reported in figure 3.26).

Finally, a lift coefficients plot is shown in figure 3.27. Results for the base mesh with 17412 elements, with and without adaptation, are compared with results obtained on the 57652 elements mesh, without adaptation. Adaptation in this case does not appear to directly improve lift coefficient results. This can be due to the ambiguous effect of the Laplacian model, which refines the mesh on the shock wave but overly stretches elements right aft of it, motivating the development of an alternative model like the mixed model presented in this chapter, not yet tested on this application.



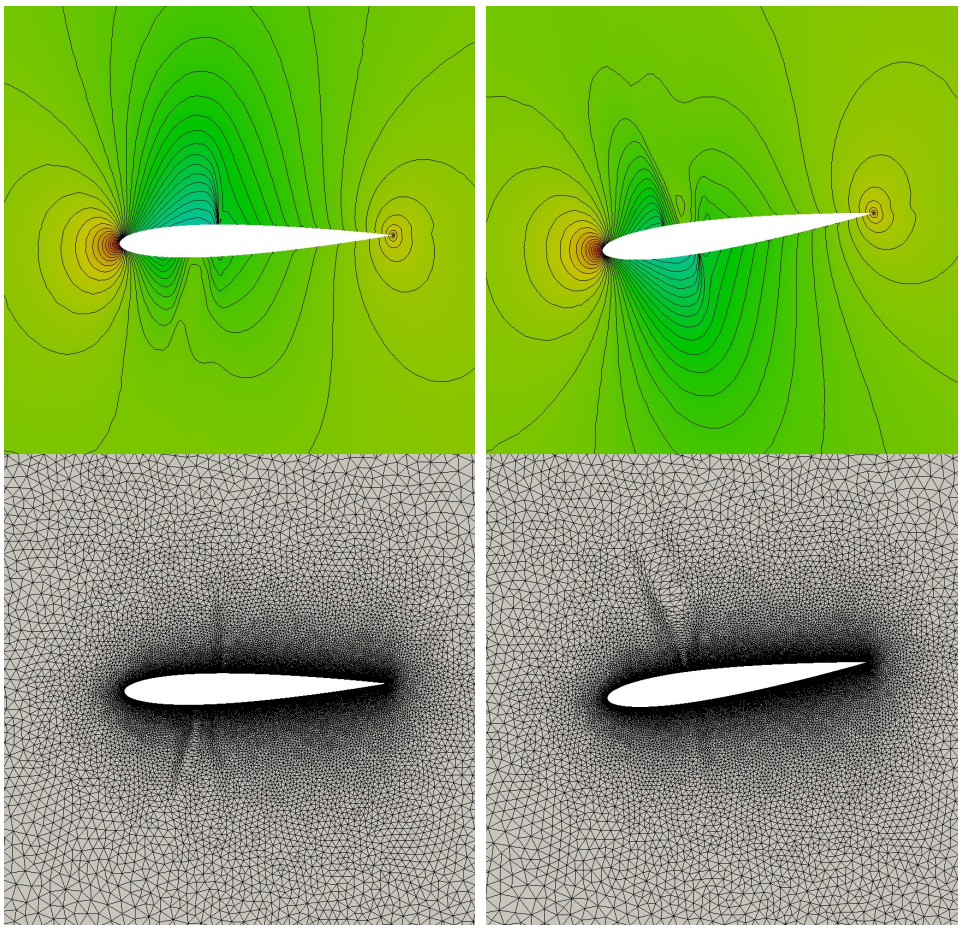
a Mass density and mesh in the steady case. b Mass density and mesh at the first time step.

Figure 3.23 – Mass density and mesh at time steps 0 and 1.



a $\alpha = 2.85°$, upstroke motion. b $\alpha = -4.966°$, minimum downstroke motion.

Figure 3.24 – Mass density and mesh at different pitching angles.



a $\alpha = 2.85°$, upstroke motion.

b $\alpha = -2.969°$, downstroke motion.

Figure 3.25 – Wavelets captured during the shock wave transition between upper and lower surface, on a more refined mesh.

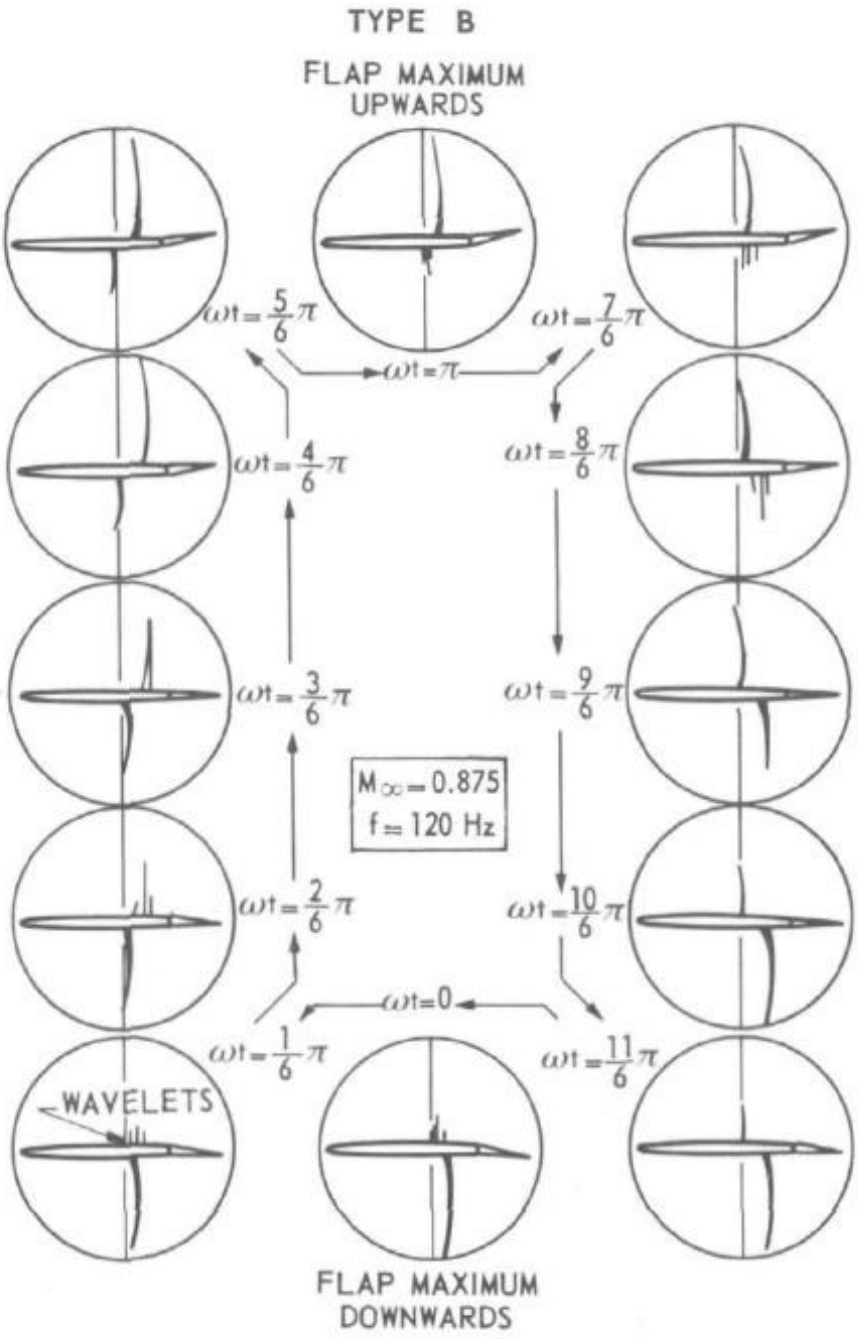


Figure 3.26 – Shock wave patterns for type B shocks (from [158], p.61.)

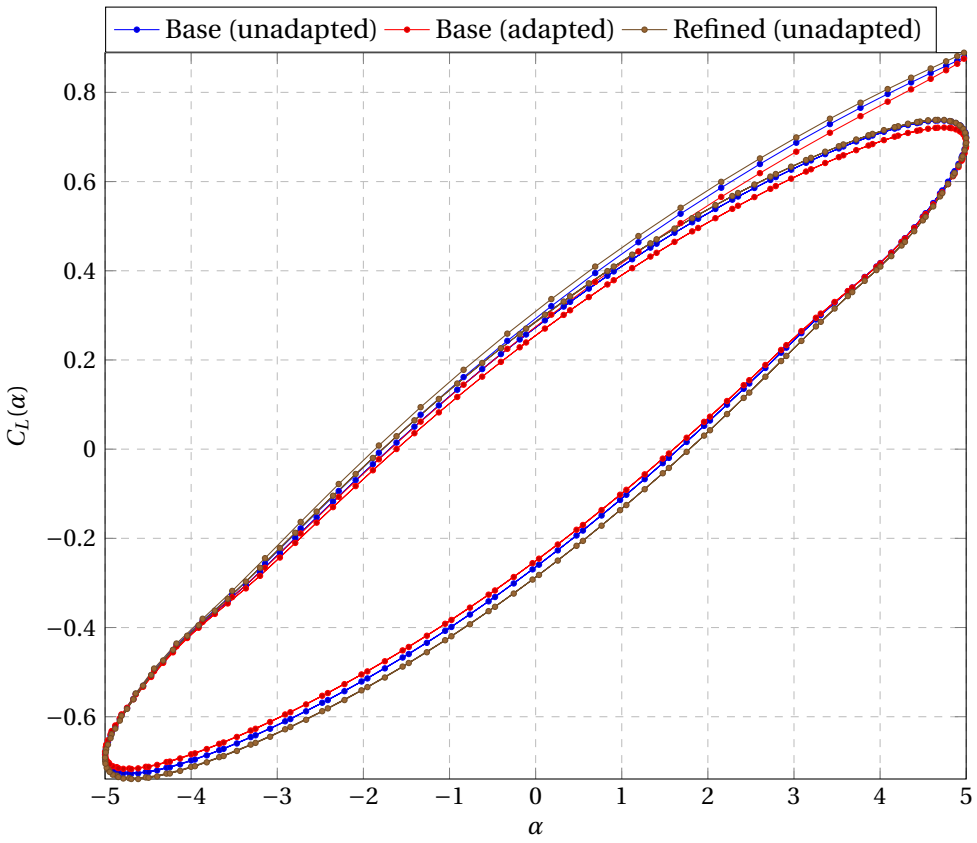


Figure 3.27 – Lift coefficient phase plot.

Part II

Numerical modeling of fluid–structure interaction and morphing

4 Rigid fluid–structure interaction and morphing boundaries

This chapter describes the novel coupling of the conservative adaptive CFD methodology described in chapter 2 with rigid body dynamics and *morphing* boundaries, i. e. boundaries change shape in time (not necessarily related to a structural displacement). The aim is to introduce the minimal additional numerical modeling which is needed to enable the aeroelastic applications which will be shown in chapter 6, namely the three-dimensional nonclassical aileron buzz simulation and the preliminary assessment of continuous wing-aileron configuration (without structural discontinuity between wing and aileron) in two dimensions. While the main numerical issue in rigid fluid structure interaction concerns the coupled time integration of the fluid flow and the bodies immersed in it, morphing boundaries mostly affect the dynamic handling of geometry parameterization, and its interaction with boundary mesh refinement. Some numerical issues related to the introduction of body elasticity, namely the transfer of displacements and loads between non-matching discretizations of the fluid and solid domains, will be introduced in chapter 5.

4.1 Partitioned solution of fluid–structure interaction problems

Simulation of fluid–structure interaction problems requires the solution of a numerical model for the fluid flow (like the ALE conservative formulation introduced in chapter 2), and a corresponding model for the structural mechanics (like rigid body dynamics or elastodynamics), provided with necessary coupling relations (as continuity of body displacements and pressure loads on the fluid–structure interface). This typically can be seen as a two-field problem, made of two continuum mechanics problems (fluid, and solid) governed by different partial differential equation and coupled through interface conditions. In principle, the two problems could be combined in a single formulation whose numerical solution is sought (the so-called *monolithic* approach). In practice, complex aeroelastic problems are mostly solved by means of *partioned* procedures [133, 56, 132] al-

lowing for a staggered time solution of the fluid and the solid problems, eventually resorting to predictor–corrector procedures to enforce numerical stability.

When a domain discretization is introduced, the need of enforcing compatibility of the fluid domain mesh with body displacement and the usage of (fictitious) continuum elasticity models for mesh deformation introduce an additional level of coupling, as the it is generally recognized that the resulting mesh volume change and velocities (needed in the ALE formulation of fluid flow) should satisfy the Geometrical Conservation Law. For this reason, fluid–structure interaction can be considered also as a *three-field problem* [56, 132], where a simultaneous solution of continuum models for fluid dynamics, solid dynamics and mesh dynamics is sought, under the constraint of interface conditions linking interface displacements of solid body and fluid mesh, interface pressure loads continuity on the fluid–solid interface, and consistency of mesh velocities with volume changes in the fluid domain and on the fluid–structure interface. In practice, this leads to the need of modifying predictor–corrector time integration schemes in order to enforce compatibility of boundary mesh velocity with structure velocity through the Geometric Conservation Law, as proposed in [57].

In the current work, the conservative ALE framework described in chapter 2 includes all mesh mechanics inside the fluid problem, avoiding potential violations of the GCL near the fluid–structure interface that could arise from a direct usage of structural velocity as mesh velocity on the interface.

4.1.1 Single-degree-of-freedom rigid body dynamics

We restrict our analysis to a single rigid body, representative of an aileron or a flap, whose only degree of freedom is the rotation β about its hinge. The motion can be than computed by means of rigid body dynamics

$$J\frac{\mathrm{d}^2\beta}{\mathrm{d}t^2} + C\frac{\mathrm{d}\beta}{\mathrm{d}t} + K\beta = \mathscr{M}^f(\mathbf{u}) \tag{4.1}$$

where J, C, K are, respectively, the rotational inertia, damping and stiffness, and M^f is the aerodynamic moment of the flap about its hinge axis (dependent on the fluid flow solution $\mathbf{u}$). Introducing the flap moment coefficient C_M^f and the time scaling $t = \hat{\mathscr{T}}\hat{\tau}$, eq. 4.1 is nondimensionalized as follows

$$\frac{J}{\hat{\mathscr{T}}^2}\frac{\mathrm{d}^2\beta}{\mathrm{d}\tau^2} + \frac{C}{\hat{\mathscr{T}}}\frac{\mathrm{d}\beta}{\mathrm{d}\tau} + K\beta = \frac{1}{2}\hat{\rho}\hat{\mathscr{L}}^d\hat{\mathscr{U}}^2 C_M^f(\mathbf{u}) \tag{4.2}$$

So

$$\underbrace{\frac{2J}{\hat{\rho}\hat{\mathscr{L}}^d\hat{\mathscr{U}}^2\hat{\mathscr{T}}^2}}_{\hat{J}}\frac{\mathrm{d}^2\beta}{\mathrm{d}\tau^2} + \underbrace{\frac{2C}{\hat{\rho}\hat{\mathscr{L}}^d\hat{\mathscr{U}}^2\hat{\mathscr{T}}}}_{\hat{C}}\frac{\mathrm{d}\beta}{\mathrm{d}\tau} + \underbrace{\frac{2K}{\hat{\rho}\hat{\mathscr{L}}^d\hat{\mathscr{U}}^2}}_{\hat{K}}\beta = C_M^f \tag{4.3}$$

where $d = 2,3$ is the number of spatial dimensions of the problem at hand (2D, 3D). A first-order system of equations in descriptor form [49] is obtained by defining the state vector

$$\mathbf{y} \triangleq \begin{bmatrix} \beta \\ \frac{\mathrm{d}\beta}{\mathrm{d}\tau} \end{bmatrix} \tag{4.4}$$

together with the system matrices and forcing

$$\mathbf{E} \triangleq \begin{bmatrix} 1 & 0 \\ 0 & \hat{J} \end{bmatrix}, \qquad \mathbf{A} \triangleq \begin{bmatrix} 0 & 1 \\ -\hat{K} & -\hat{C} \end{bmatrix}, \qquad \mathbf{f}(\mathbf{u}) \triangleq \begin{bmatrix} 0 \\ C_M^f(\mathbf{u}) \end{bmatrix} \tag{4.5}$$

so that the system reads

$$\mathbf{E}\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\tau} = \mathbf{A}\mathbf{y} + \mathbf{f}(\mathbf{u}) \tag{4.6}$$

While coupling a Computational Fluid Dynamics (CFD) solver with a Computational Solid Dynamics solver can be done both by the implementation of monolithic solvers (solving both the fluid flow equations and the rigid/elastic solid dynamics equations at the same time, as for example in [119]) and partitioned ones (solving the two fields one at a time, in a staggered pattern [56, 133, 132]), we restrict our attention to the second approach, since it allows the least invasive coupling of the existing flow solver (chapter 2) with the rigid body motion.

4.1.2 predictor–corrector coupled time integration of fluid flow and rigid body dynamics

At each time step, the aerodynamic forcing in system 4.6 depends on the fluid flow around the flap at angle β. A coupled time integration procedure is thus needed for the fluid flow and flap dynamics. A convenient staggered solution of the fluid flow and flap motion can be achieved through the adoption of a predictor–corrector scheme. We use the predictor–corrector scheme proposed by Giles [73] (and used for aeroelastic applications in [40, 35, 33])

$$\begin{aligned}
&\text{Predictor:} && \left(\mathbf{E} - \frac{\Delta\tau}{2}\mathbf{A}\right)\mathbf{y}^* = \left(\mathbf{E} + \frac{\Delta\tau}{2}\mathbf{A}\right)\mathbf{y}^{(n)} + \Delta\tau\mathbf{f}^{(n)} \\
&\text{Evaluation:} && \mathbf{f}^{(n+1)} = \mathscr{F}(\mathbf{y}^*) \\
&\text{Corrector:} && \mathbf{E}\mathbf{y}^{(n+1)} = \mathbf{E}\mathbf{y}^{(n)} + \frac{\Delta\tau}{2}\mathbf{A}\left(\mathbf{y}^{(n)} + \mathbf{y}^*\right) + \frac{\Delta\tau}{2}\left(\mathbf{f}^{(n)} + \mathbf{f}^{(n+1)}\right)
\end{aligned} \tag{4.7}$$

At each time step, the predicted state $\mathbf{y}^*$ is computed by means of the trapezoidal rule, with an explicit treatment of the forcing $\mathbf{f}$. Then, the forcing $\mathbf{f}^{(n+1)}$ is evaluated based on the fluid flow solution $\mathbf{u}$ on the mesh compatible with the predicted state $\mathbf{y}^*$, and finally a correction step is applied to get $\mathbf{y}^{(n+1)}$

The predicted state $\mathbf{y}^*$ is used as a boundary condition for mesh deformation, according to the procedure described in 2.5. When the mesh deformation becomes difficult, the time step is split according to the algorithm 2. Thus, multiple predictor steps are performed as follows

$$\begin{aligned} \mathbf{y}_0^* &= \mathbf{y}^{(n)} \\ \left(\mathbf{E} - \frac{\Delta\tau_k}{2}\mathbf{A}\right)\mathbf{y}_{k+1}^* &= \left(\mathbf{E} + \frac{\Delta\tau_k}{2}\mathbf{A}\right)\mathbf{y}_k^* + \Delta\tau_k \mathbf{f}^{(n)}, \qquad k = 0, \dots, K-1 \end{aligned} \tag{4.8}$$

Correction is performed only at the end of the time step, when it is actually possible to evaluate the current aerodynamic loads.

4.1.3 predictor–corrector assessment on a model problem

We test the performance of the predictor–corrector scheme on a model problem with proportional forcing

$$\mathbf{E} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad \mathbf{A} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \qquad \mathbf{f}(\mathbf{y}) = \begin{bmatrix} 0 & 0 \\ -\kappa^2 & 0 \end{bmatrix}\mathbf{y} \tag{4.9}$$

with initial conditions

$$\beta(0) = 0, \qquad \frac{\mathrm{d}\beta}{\mathrm{d}\tau}(0) = \kappa B \tag{4.10}$$

This is equivalent to a staggered solution of the unforced oscillator

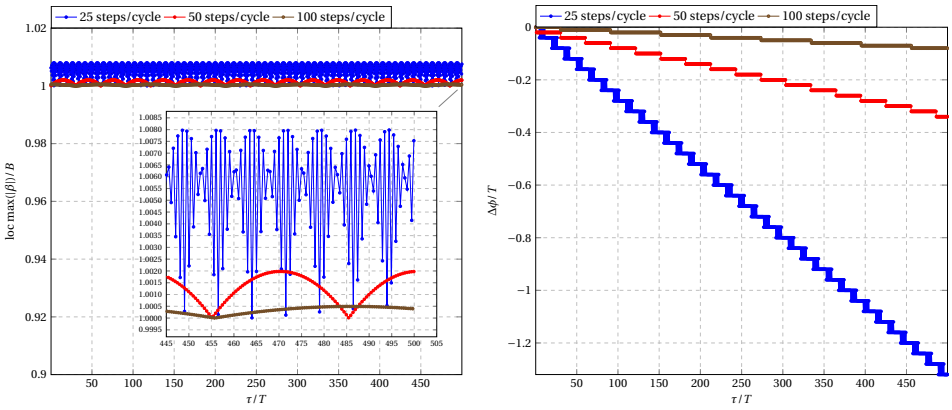
$$\frac{\mathrm{d}^2\beta}{\mathrm{d}\tau^2} + \kappa^2\beta = 0, \qquad \text{s.t.} \beta(0) = 0, \quad \frac{\mathrm{d}\beta}{\mathrm{d}\tau}(0) = \kappa B \tag{4.11}$$

whose analytical solution is the periodic solution with pulsation κ and amplitude B

$$\beta(\tau) = B\sin(\kappa\tau) \tag{4.12}$$

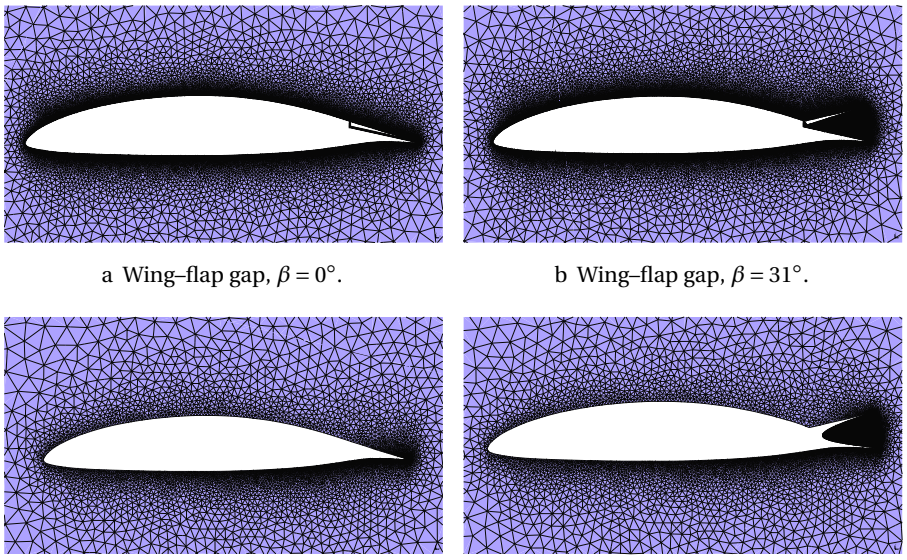
We do not attempt here to derive an analytical proof of the stability of the predictor–corrector time integration scheme derived in section 4.1.2 when applied to the model problem 4.9. Instead, we verify it numerically for increasing values of $\Delta\tau$, and we perform numerical tests to assess the effects of the time discretization on the frequency error.

Figure 4.1a shows the variations in the absolute local maxima loc max($|\beta|$) relative to the amplitude B, with respect to simulation time τ (rescaled by period $T = 2\pi/\kappa$). We can see that, for this model problem, the amplitude error remains bounded below 1% of the amplitude even with a coarse time discretization with 25 points per period. Figure 4.1b shows the time lag $\Delta\phi$ between the local maxima in the simulated signal and in the original one. The time lag appears to be linear, and for the coarsest time discretization the simulated signal anticipates the analytical one of 1 period every 400 periods simulated. A medium time discretization of 50 points per periods already reduces this lag below 30% of a period every 400 periods simulated.



a Absolute local maxima. b Time lag in local maxima.

Figure 4.1 – Absolute local maxima variation and their time shift for different time discretizations.



a Wing–flap gap, $\beta = 0°$. b Wing–flap gap, $\beta = 31°$.

c Continuous wing–flap, $\beta = 0°$. d Continuous wing–flap, $\beta = 31°$.

Figure 4.2 – Example of discontinuous and continuous wing–flap configurations on a HQ17 airfoil.

4.2 Morphing boundaries over adaptive meshes

Wing–flap configurations are mostly studied as two different rigid or elastic bodies, separated by a thin air gap. We would like to explore the effect of the air gap on shock wave motion and on aeroelastic stability by removing it and replacing it

with a continuous surface connection (figure 4.2). Modeling this connection surface during an unsteady simulation with relative movement between airfoil and flap creates the necessity of parameterizing the motion of the surface so that it always remains attached to the moving bodies, while keeping a desired level of smoothness. We typically know an initial position of the surface $\mathbf{x}^{(0)} \in \mathbb{R}^d$, and possibly some other intermediate configurations or the final one $\mathbf{x}^{(T)} \in \mathbb{R}^d$. Since we are only interested in creating a continuous surface, without entering into the details of the structural modeling of the two bodies, a detailed elastic modeling of the surface is not the only solution to this *shape morphing* problem, which is common to other research fields such as industrial design [39] and computer graphics [107, 10, 177]. Three steps have to be considered:

1. Solution of a *correspondence problem*, with the definition of a common parametric coordinate $\boldsymbol{\xi} \in \mathbb{R}^{d-1}$ used to relate the initial and final configurations $\mathbf{x}^{(0)}(\boldsymbol{\xi})$ and $\mathbf{x}^{(T)}(\boldsymbol{\xi})$ in a Lagrangian fashion.

2. A *trajectory problem* giving the parametrical shape variation, is solved.

3. In the present application, *interaction with mesh adaptation* needs to be considered in order to decide how to include new mesh nodes into the time evolution of the surface (without spoiling surface smoothness).

Several alternative approaches for the solution of the trajectory problem have been considered in this work.

- **Variational models** rely on partial differential equations models to describe the shape trajectory $\mathbf{x}(\boldsymbol{\xi}, t)$. In the present application, a first candidate for this role is of coarse a *dynamic elastic model*, where the morphing process is driven by the forcing terms. Defining the displacement $\mathbf{u} = \mathbf{x} - \mathbf{x}^{(0)}$, the nonlinear dynamic elastic model reads [135]

$$\int_\Omega \delta\mathbf{u} \cdot \left(\rho_0 \frac{\partial^2 \mathbf{u}}{\partial t^2}\right) \mathrm{d}\Omega + \int_\Omega \delta\epsilon(\mathbf{u}) : \sigma(\mathbf{u})\, \mathrm{d}\Omega = \int_\Omega \rho_0 \delta\mathbf{u} \cdot \mathbf{f}(t)\, \mathrm{d}\Omega + \int_{\partial\Omega} \delta\mathbf{u} \cdot \mathbf{n} P(t)\, \mathrm{d}\Gamma \tag{4.13}$$

  Membrane elements seem to be the simplest choice for surface discretization. However, from the practical point of view this choice poses two problems:

  1. How to choose a time-varying forcing so that a monotonic smooth movement is ensured?

  2. The moving membrane should never intersect other boundaries.

For this reasons, a dynamic structural model has not been implemented in the current work. For the purpose of geometrical modeling, it could be better to set aside a rigorous physical interpretation for the surface movement, and to look for a simpler model which could provide smooth results. A good solution to the second issue is the combination of a variational model with shape interpolation. Poisson equation models [177] have been proposed in order to minimize the $H^1(\Omega)$ norm of the actual shape $\mathbf{x}(\boldsymbol{\xi})$ and a target one $\mathbf{y}(\boldsymbol{\xi})$

$$\underset{\mathbf{x}}{\operatorname{argmin}} ||\mathbf{x}-\mathbf{y}||_{H^1(\Omega)} \tag{4.14}$$

leading to the variational formulation

$$\int_\Omega \boldsymbol{\nabla}\mathbf{v}\cdot\boldsymbol{\nabla}\mathbf{x}\,\mathrm{d}\Omega = \int_\Omega \boldsymbol{\nabla}\mathbf{v}\cdot\boldsymbol{\nabla}\mathbf{y}\,\mathrm{d}\Omega \tag{4.15}$$

In this way, only gradients of the target shape need to be prescribed, and they can be provided through an interpolation formula

$$\boldsymbol{\nabla}\mathbf{y}(\boldsymbol{\xi},t) = \mathbf{F}(t)\boldsymbol{\nabla}\mathbf{x}^{(0)} + \mathbf{G}(t)\boldsymbol{\nabla}\mathbf{x}^{(T)} \tag{4.16}$$

where the two matrices $\mathbf{F}(t)$ and $\mathbf{G}(t)$ allow for a possibly nonlinear interpolation [177].

This approach is particularly simple for complex geometrical models, where much of the burden is in the solution of the correspondence problem between the initial and final surface mesh, but solution of a Poisson equation is quite straightforward. Changing the minimization norm gives the possibility to change the PDE model. For example, using the elastic energy norm would give the possibility of driving the model through the interpolation of stresses on the initial and final configuration, linking the dynamic elastic problem with the variational shape interpolation technique proposed in [177]. Anyway, for a successful coupling with mesh adaptation, we are interested in preserving surface smoothness as much as possible. Looking for a solution in H^1 could be done with linear finite elements, which are only C^0 globally. When this is coupled with mesh adaptation, new nodes would be added on straight lines (according to an isoparametric representation), instead of querying a geometrical model, thus spoiling the smoothness of the original surface. Moreover, adaptive finite element discretization of the variational model would be needed for handling the variable surface discretization produced by mesh adaptation.

- **Direct geometrical model parameterization**, instead, offers the possibility of preserving the desired curve smoothness along time, by introducing a

geometrical model of the initial and target configurations and directly interpolating in time the geometric model coefficients. Although this approach could be problematic for complex geometrical models, for the present two-dimensional application it is extremely simple to define the same parametric coordinate $\xi \in [0,1]$ for the initial and final shapes, thus solving the correspondence problem, and to introduce a geometrical model for the curve in $\mathbb{R}^2$. This approach will be followed in section 4.2.1, while shape interpolation will be introduced in section 4.2.2, and interaction with mesh adaptation will be considered in section 4.2.3.

4.2.1 Geometry parameterization

A possible boundary geometry modelling of two-dimensional initial and target configurations $\mathbf{x}^{(0)}(\xi)$, $\mathbf{x}^{(T)}(\xi)$, parameterized on the reference domain $\xi \in [0,1]$, is given by a composite cubic Bézier approximation

$$\begin{aligned} \mathbf{x}^{(0)}(\xi) &= \sum_{i=1}^{N-1} \mathbf{K}_i^{(0)} \mathbf{b}_i(\xi), \qquad \xi \in [0,1], \quad i = 1,\dots,N-1 \\ \mathbf{x}^{(T)}(\xi) &= \sum_{i=1}^{N-1} \mathbf{K}_i^{(T)} \mathbf{b}_i(\xi), \qquad \xi \in [0,1], \quad i = 1,\dots,N-1 \end{aligned} \tag{4.17}$$

where Bézier curves $\mathbf{b}_i(\xi)$ are locally defined on subdomains

$$[\xi_i, \xi_{i+1}] \subset [0,1], \quad i = 1,\dots,N-1 \tag{4.18}$$

A shape interpolation procedure for the boundary position $\mathbf{r}(\xi,t)$ can be introduced as

$$\mathbf{x}(\xi,t) = \mathbf{F}(t)\mathbf{x}^{(0)}(\xi) + \mathbf{G}(t)\mathbf{x}^{(T)}(\xi), \qquad \xi \in [0,1], \quad t \in [0,T] \tag{4.19}$$

subject to the interpolation conditions

$$\begin{aligned} \mathbf{F}(0) &= \mathbf{I}, \qquad \mathbf{F}(T) = \mathbf{0} \\ \mathbf{G}(0) &= \mathbf{0}, \qquad \mathbf{G}(T) = \mathbf{I} \end{aligned} \tag{4.20}$$

The only dependence on time is retained by the interpolation matrices $\mathbf{F}(t)$, $\mathbf{G}(t)$. Introducing the same geometric model for the curve position $\mathbf{x}(\xi,t)$

$$\mathbf{x}(\xi,t) = \sum_{i=1}^{N-1} \mathbf{K}_i(t) \mathbf{b}_i(\xi), \qquad \xi \in [0,1], \quad t \in [0,T] \tag{4.21}$$

we get a the interpolation formula for the discretized shape

$$\mathbf{K}_i(t) = \mathbf{F}(t)\mathbf{K}_i^{(0)} + \mathbf{G}(t)\mathbf{K}_i^{(T)}, \qquad i = 1,\dots,N-1, \quad t \in [0,T] \tag{4.22}$$

which is directly expressed in terms of the geometric model coefficients $\mathbf{K}_i(t)$.

4.2.2 Nonlinear shape interpolation

Since we are interested in a continuous surface connecting two rigid bodies subject to a relative rotation, a nonlinear shape interpolation is needed to make the surface comply with the flap rotation at each time instant.

The flap tip position can be written as its rotation by matrix $\Delta\mathbf{R}(\beta(t))$ about the flap hinge located at $\mathbf{h}$

$$\mathbf{x}_{\text{tip}}(t) = \mathbf{h} + \Delta\mathbf{R}(\beta(t))(\mathbf{x}_{\text{tip}}^{(0)} - \mathbf{h}) \tag{4.23}$$

Compatibility of shape interpolation with flap tip relative rotation gives

$$\mathbf{x}_{\text{tip}}(t) = \mathbf{F}(t)\mathbf{x}_{\text{tip}}^{(0)} + \mathbf{G}(t)\mathbf{x}_{\text{tip}}^{(T)} = \mathbf{h} + \Delta\mathbf{R}\left(\beta(t)\right)(\mathbf{x}_{\text{tip}}^{(0)} - \mathbf{h}) \tag{4.24}$$

By explicitating the hinge position $\mathbf{h}$ with respect to the initial and final flap tip positions we get the compatibility condition

$$\begin{aligned}\mathbf{x}_{\text{tip}}(t) = &\left(\Delta\mathbf{R}\left(\beta(t)\right) - \left(\mathbf{I} - \Delta\mathbf{R}\left(\beta(t)\right)\right)\left(\mathbf{I} - \Delta\mathbf{R}^{(T)}\right)^{-1}\Delta\mathbf{R}^{(T)}\right)\mathbf{x}_{\text{tip}}^{(0)} + \\ &+ \left(\mathbf{I} - \Delta\mathbf{R}\left(\beta(t)\right)\right)\left(\mathbf{I} - \Delta\mathbf{R}^{(T)}\right)^{-1}\mathbf{x}_{\text{tip}}^{(T)}\end{aligned} \tag{4.25}$$

One possible way to choose an explicit expression of $\mathbf{F}(t)$ and $\mathbf{G}(t)$ so that they comply with flap rotation is to define them as

$$\begin{aligned}\mathbf{G}(t) &\triangleq \left(\mathbf{I} - \Delta\mathbf{R}\left(\beta(t)\right)\right)\left(\mathbf{I} - \Delta\mathbf{R}^{(T)}\right)^{-1} \\ \mathbf{F}(t) &\triangleq \Delta\mathbf{R}\left(\beta(t)\right) - \mathbf{G}(t)\Delta\mathbf{R}^{(T)}\end{aligned} \tag{4.26}$$

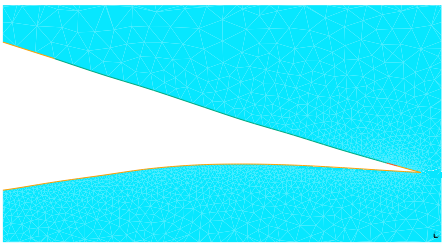
It can be verified that this choice automatically fulfils the interpolation conditions

$$\begin{aligned}\mathbf{F}(0) &= \mathbf{I}, \qquad \mathbf{F}(T) = \mathbf{0} \\ \mathbf{G}(0) &= \mathbf{0}, \qquad \mathbf{G}(T) = \mathbf{I}\end{aligned} \tag{4.27}$$

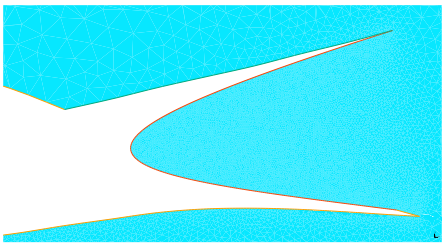
It should be noted that, altought the present interpolation of the geometrical model guarantees C^2 continuity $\forall\xi \in (0,1)$, the current choice of the interpolation matrices only preserves C^0 smoothness at end points. This issue is in fact shared by every shape morphing model considered in this section, as also Poisson and elastic models only allow the mutually exclusive imposition of Dirichlet or Neumann boundary conditions at end points. At least, the current model allows to be flawlessly coupled to mesh adaptation, while preserving the prescribes surface smoothness.

This choice automatically allows a free rotation of the flap under the effects of the aerodynamic loads, as the flap angle $\beta(t)$ is used to define the interpolation matrices at each time instant.

Figure 4.3 shows the initial and target configurations for a HQ17 airfoil, while figure 4.5 shows the results of the nonlinear shape interpolation procedure.



a Continuous wing–flap, $\beta = 0°$.



b Continuous wing–flap, $\beta = 31°$.

Figure 4.3 – Initial and target configurations for the continuous flap opening on the HQ17 airfoil.

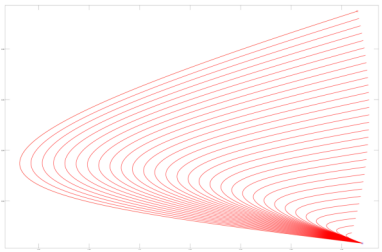


Figure 4.4 – Continuous wing–flap, interpolated shapes for $\beta \in [0°, 31°]$.

Figure 4.5 – Interpolated configurations for the HQ17 airfoil.

4.2.3 Interaction with mesh adaptation

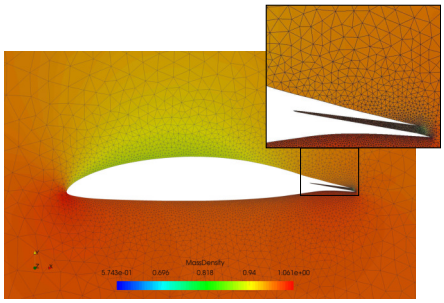
Flawless interaction of the nonlinear shape interpolation procedure with dynamic mesh adaptation relies on the distinction of curve *control points* from *evaluation points*. Curve *knots/control points* are considered separately from boundary mesh points. Having defined the same composite cubic Bézier model for all the curve shapes in the time interval $t \in [0, T]$ means that the number of spline control points remains constant in time. Boundary mesh points are *evaluation points* of the geometric model, thus each time that mesh adaptation inserts a new node on the boundary, its parametric coordinate ξ is guessed from the average of the parametric coordinates of its neighbors, then the position is corrected by evaluating the geometrical model at ξ.

4.2.4 Example: Partial and complete flap opening

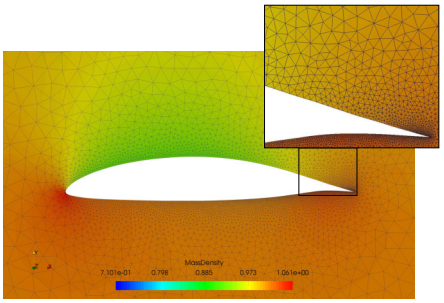
A preliminary application, intended for model testing purposes and not physical results validation, is the partial and complete flap opening on a HQ17 airfoil in a subsonic flow at $M_\infty = 0.35$, $\rho_\infty = 1.4$, at an angle of attack $\alpha = 3°$. In the partial opening case, flap angle varies from $\beta = 3°$ to $\beta = 31°$, starting from an already

open flap configuration, continuously connected to the airfoil upper surface. In the full opening case, flap angle varies from $\beta = 0^\circ$ to $\beta = 28^\circ$, starting from a clean airfoil configuration with no flap extending outside of the upper airfoil surface. This is quite an extreme test case, intended for an algorithmic validation of the geometrical model before its application to a physical problem, the nonclassical aileron buzz in chapter 6.

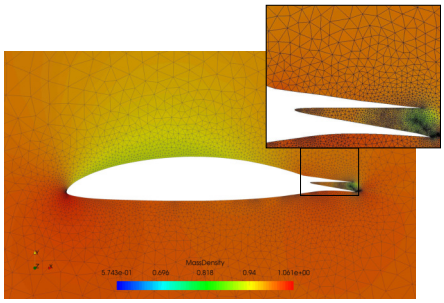
Figure 4.6 shows some time snapshots for the partial and full opening flap cases. Mesh adaptation is performed and is necessary to maintain a good quality body-fitted mesh, although we do not attempt in this test case to target specific flow features.



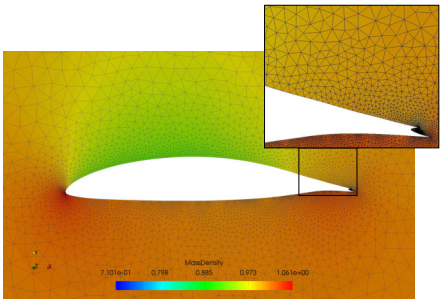
a Partial opening, $\beta = 3°$.



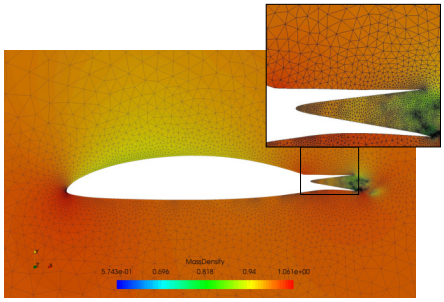
b Full opening, $\beta = 0°$.



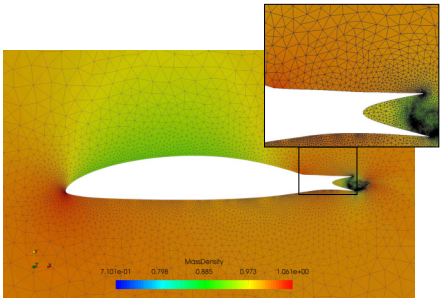
c Partial opening, $\beta = 10°$.



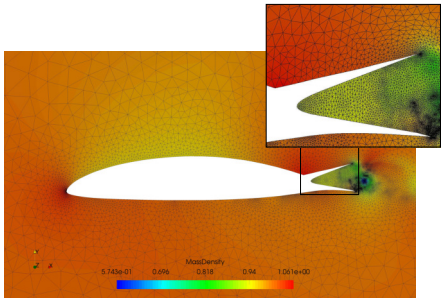
d Full opening, $\beta = 7°$.



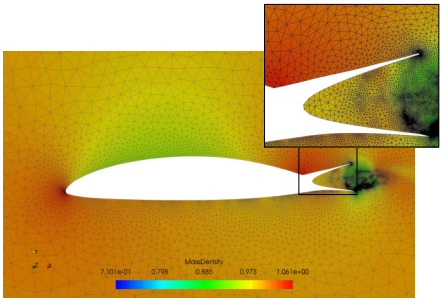
e Partial opening, $\beta = 17°$.



f Full opening, $\beta = 14°$.



g Partial opening, $\beta = 31°$.



h Full opening, $\beta = 28°$.

Figure 4.6 – Mass density distribution for partial and full flap opening at $M = 0.35$, with nonlinear shape interpolation and mesh adaptation.

5 Generalized beam models for camber-morphing aeroelastic applications

This chapter introduces the last contributions of this work to the topics of structural dynamics and morphing. The main focus is on structural modeling, seen as a separable (yet not independent) component of an aeroelastic simulation, with the development of a low-dimensional model specific for wings geometries amenable to generalized beam model analysis.

Section 5.1 introduces the topic of structural morphing and generalized beam models. Section 5.2 shows how a cross-section discretization is used to derive an ODE formulation of the elastic problem along the beam axis, while section 5.3 presents the projection-based approach leading to the fully discrete formulation shown in section 5.4. Nondimensionalization and validation of the model are introduced in sections 5.5 and 5.6, respectively, while application to camber-morphing wing sections is shown in section 5.7

Coupling with a fluid flow solver through an interface algorithm, and the specific issues which have limited its application in this work, are finally presented in section 5.8.

5.1 Introduction

What shape morphing is *Morphing* is a verb which is derived from the abbreviation of *metamorphosis*. *Shape morphing* refers to a continuous change in the external shape of an object, and in the aeronautical community it denotes the possibility of changing the shape of some aircraft components during flight. This can be done in order to implement an actuation systems, or to optimize performances in different flight conditions, just to give some examples of possible applications.

The external shape of lifting surfaces of an aircraft is typically designed so to fulfill some desired aerodynamic performance in a specified flight condition. Even if the possibility of continuously changing wing shape during flight was present in the earliest aircraft design (such as the wing twisting mechanism on the Wright Flyer [15]), this feature has disappeared in modern aircraft as the increasingly higher flight speed led to the discovery of aeroelastic instability phenomena [26] demanding for stiffer aircraft structures. Classical aircraft structure design has

been developed based on the usage of stiff materials and topologies to accomplish the load bearing role, with a limited shape-changing capability localized in control surfaces such as flaps, aileron, spoilers, etc.

Morphing aircraft structures, i.e. structures capable of flexible seamless changes of shape, have been introduced in the last years for their potential capability of adaptation with a limited weight and reliability penalty [15, 168]. Shape changes on wings can include variations in span, twist, camber [15]. Among all possible shape modification, in this work we focus on camber morphing, whose application on rotorcraft blades offers a significant room for optimization of performance of the vehicle, included the reduction of vibratory loads and emitted noise [67, 122, 178].

Structural design, material design, simulation A large change of the design paradigms is required to develop structural concepts that can fulfil effectively, at the same time, the two roles of load bearing and shape morphing. Examples of possible innovative configurations are given in [4, 143]. Most of the works available in the literature either focus on the design of *smart* materials capable of withstanding the shape change process, or on the design of the actuation system, or on the topological optimization of the structure. In order to be able to perform a computer simulation of an aeronautical morphing system, the combination of a structural elastic model, an aerodynamic model, and possibly a model of the actuation system is required. The usage of state-of-the-art computational methods (such as finite elements) for each component of this multidisciplinary framework would lead to a complex high-dimensional model, that would require HPC capabilities in order to run the simulations.

In order to develop a computational model that could be competitive (in terms of time and computational resources) with traditional aeronautical design methods, in this work we explore the capabilities of generalized beam models to deal with camber morphing of straight, slender wings.

Generalized beam models Classical beam models are often analytically derived from three-dimensional continuum mechanics by introducing some additional hypotheses on the kinematic behaviour of the cross-section of long, prismatic solid bodies [159, 135]. Several works have focused on generalizing beam models by means of semi-analytical methods providing a detailed characterization of the beam cross-section (summarized in [83]), mostly based on a finite element modelling of the cross-section [72, 83], allowing for composite and anisotropic materials. In [72], a decomposition of the displacement field into a particular integral made of a polynomial solution (*central* solution field) and a general integral made of a set of self-balanced exponential solution (*warping* solution field)

is introduced, without any a priori hypothesis on the cross-section kinematic behaviour. The first part of the solution is determined by the applied load, and extends the classical beam polynomial solution by allowing for some section deformation, while the second part recovers the local deformation introduced by the boundary conditions, which is usually neglected in engineering applications. The exponential decay of the warping field is in accordance with de Saint Venant's principle [85, 84]).

In [125], it has been shown that the same solutions can be obtained without assumptions on the section behaviour, and without any a priori displacement decomposition, once the continuum linear elasticity problem is reformulated as an evolutionary problem along the beam axis [125, 72, 120] and the eigenanalysis of the resulting first order system of linear Ordinary Differential Equations (ODEs) is studied. This reformulation takes advantage only of an in-plane discretization of the solid cross-section (typically a finite-element one), and generates a system of ODEs which is twelve times singular. The six rigid displacements and the six classical polynomial solutions for the beam displacement automatically appear as the solutions generated by the generalized eigenfunctions associated to null eigenvalues. Additional displacement fields are generated by eigenfunctions associated to non null eigenvalues, as in the decoupling proposed in [72].

The aim of this work is to use this formulation to identify a set of solutions clearly related to in-plane deformation, in order to derive generalized beam models capable of representing the camber morphing behaviour of airfoil sections. The formulation presented in [125] starts from the complete three-dimensional elastic continuum to derive an approximate three-dimensional solution made of a combination of a reduced number of eigenfuctions. This can be alternatively be interpreted as a projection of the full-order finite element model on a reduced number of basis functions, as it is done in projection-based model reduction [12]. This interpretation allows to investigate also choices of the basis functions different from the system eigenfunctions, while retaining the same computational framework, and will be adopted in the following.

5.2 Semi-discrete formulation of linear elastic mechanics

We consider a prismatic three-dimensional elastic solid, with constant cross-section and straight out-of-plane x-axis (fig. 5.1). In-plane coordinates are labeled as $\boldsymbol{\xi} \triangleq (y, z)$.

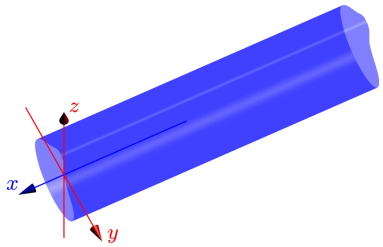


Figure 5.1 – Constant cross-section prismatic solid.

5.2.1 Virtual work principle

The starting point is the virtual work principle for a linear elastic continuum [135], stating the equality of the internal and external virtual work in the three-dimensional domain Ω occupied by the elastic solid

$$\delta\mathscr{L}_i = \delta\mathscr{L}_e \tag{5.1}$$

for every suitable virtual displacement field $\delta\mathbf{u}$.

The internal virtual work is expressed in terms of the stress tensor $\sigma(\mathbf{u})$ and the compatible variation of the strain tensor $\delta\mathbb{e} = \mathbb{e}(\delta\mathbf{u})$

$$\delta\mathscr{L}_i = \int_\Omega \mathbb{e}(\delta\mathbf{u}) : \sigma(\mathbf{u})\,\mathrm{d}\Omega \tag{5.2}$$

while the virtual external work, considering a cantilever constant cross-section prism of length L in the interval $x \in [0, L]$, can be written as

$$\delta\mathscr{L}_e = \int_\Omega \delta\mathbf{u}\cdot\mathbf{f}_\Omega\,\mathrm{d}\Omega + \int_{\mathscr{S}_L} \delta\mathbf{u}\cdot\mathbf{f}_L\,\mathrm{d}\mathscr{A} + \int_{\partial\Omega\setminus\{\mathscr{S}_0\cup\mathscr{S}_L\}} \delta\mathbf{u}\cdot\boldsymbol{\tau}\,\mathrm{d}\Gamma \tag{5.3}$$

where $\mathbf{f}_\Omega$ represent volume forces, $\mathbf{f}_L$ represents the surface forces applied on the end section $\mathscr{S}_L$ (assuming Dirichlet conditions on the root section $\mathscr{S}_0$), and $\boldsymbol{\tau}$ represents the stress applied on the lateral surface $\partial\Omega\setminus\{\mathscr{S}_0\cup\mathscr{S}_L\}$.

The linear elastic material is characterized by the constitutive law

$$\sigma(\mathbf{u}) = 2\mu\mathbb{e}(\mathbf{u}) + \lambda\mathrm{tr}(\mathbb{e}(\mathbf{u})) \tag{5.4}$$

and the infinitesimal deformations hypothesis leads to the adoption of the small strain tensor

$$\mathbb{e}(\mathbf{u}) = \frac{1}{2}\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right) \tag{5.5}$$

The symmetric stress and small strain tensors can be conveniently re-arranged as six-component arrays $\boldsymbol{\sigma}, \boldsymbol{\epsilon}$, the latter related to the displacement field $\mathbf{u}$ by

means of a linear differential operator $\mathscr{D}$ through the relation

$$\boldsymbol{\epsilon} = \mathscr{D}\mathbf{u} = \begin{bmatrix} \partial/\partial x & 0 & 0 \\ 0 & \partial/\partial y & 0 \\ 0 & 0 & \partial/\partial z \\ \partial/\partial y & \partial/\partial x & 0 \\ \partial/\partial z & 0 & \partial/\partial x \\ 0 & \partial/\partial z & \partial/\partial y \end{bmatrix} \mathbf{u} \tag{5.6}$$

so that the linear elastic constitutive law reads

$$\boldsymbol{\sigma} = \mathbf{D}\boldsymbol{\epsilon} \tag{5.7}$$

where

$$\mathbf{D} = \begin{bmatrix} 2\mu+\lambda & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & 2\mu+\lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & 2\mu+\lambda & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu & 0 & 0 \\ 0 & 0 & 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu \end{bmatrix} \tag{5.8}$$

This allows us to compactly write the virtual work principle as

$$\boxed{\int_\Omega \boldsymbol{\epsilon}(\delta\mathbf{u})^T \mathbf{D}\boldsymbol{\epsilon}(\mathbf{u})\,\mathrm{d}\Omega = \delta\mathscr{L}_e, \qquad \forall\delta\mathbf{u}} \tag{5.9}$$

5.2.2 Separation of variables and cross-section discretization

In order to obtain an evolutionary form of the semi-discrete of the elastic problem along the space direction x, it is convenient to split the strain tensor $\boldsymbol{\epsilon}$ into the contributions brought by the derivatives in the in-plane directions $\boldsymbol{\xi} \triangleq (y,z) \in \mathscr{S}$ and the derivative in the beam axis direction $x \in [0, L]$

$$\boldsymbol{\epsilon} = \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & \partial/\partial y & 0 \\ 0 & 0 & \partial/\partial z \\ \partial/\partial y & 0 & 0 \\ \partial/\partial z & 0 & 0 \\ 0 & \partial/\partial z & \partial/\partial y \end{bmatrix}}_{\triangleq \mathscr{D}_{\boldsymbol{\xi}}} \mathbf{u} + \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}}_{\triangleq \mathbf{S}} \frac{\partial \mathbf{u}}{\partial x} = \mathscr{D}_{\boldsymbol{\xi}}\mathbf{u} + \mathbf{S}\frac{\partial \mathbf{u}}{\partial x} \tag{5.10}$$

At this point, we can introduce a discretization for the cross-section domain $\mathscr{S}$ (which is constant along the x-axis)

$$u^d(\boldsymbol{\xi}, x) = \sum_{s=1}^{N_s} \phi_s(\boldsymbol{\xi})\, v_s^d(x), \qquad d = 1,..,3 \tag{5.11}$$

In this work, ϕ_s is a nodal P^1 linear finite element basis function, so that $v_s^d(x)$ represents the displacement of node s in the space direction d. A convenient matrix re-arrangement of the last expression reads

$$\mathbf{u}(\boldsymbol{\xi}, x) = \mathbf{N}(\boldsymbol{\xi})\mathbf{v}(x) \tag{5.12}$$

Substitution into the expression of the strain tensor splitting (eq. 5.10) gives

$$\boldsymbol{\epsilon} = \underbrace{\mathscr{D}_{\boldsymbol{\xi}}\mathbf{N}(\boldsymbol{\xi})}_{\triangleq \mathbf{Z}_{\boldsymbol{\xi}}(\boldsymbol{\xi})}\mathbf{v}(x) + \underbrace{\mathbf{S}\mathbf{N}}_{\triangleq \mathbf{Z}_0}\frac{\mathrm{d}\mathbf{v}(x)}{\mathrm{d}x} = \mathbf{Z}_{\boldsymbol{\xi}}(\boldsymbol{\xi})\mathbf{v}(x) + \mathbf{Z}_0\frac{\mathrm{d}\mathbf{v}(x)}{\mathrm{d}x} \tag{5.13}$$

Thus, the strain tensor $\boldsymbol{\epsilon}$ is directly expressed in terms of the section nodal displacements $\mathbf{v}(x)$ (since in-plane derivatives are computed analytically for the P^1 finite element basis) and their first derivative $\mathbf{v}'(x) = \frac{\mathrm{d}\mathbf{v}}{\mathrm{d}x}$ along the beam axis.

Since only matrix $\mathbf{Z}_{\boldsymbol{\xi}}(\boldsymbol{\xi})$ retains a dependence from the cross-section coordinates, the last expression allows us to express the volume integral in the virtual internal work (eq. 5.9) as a multiple integral over the surface $\mathscr{S}$ and the line $[0, L]$, getting

$$\begin{aligned}\delta\mathscr{L}_i &= \int_\Omega \boldsymbol{\epsilon}(\delta\mathbf{u})^T\mathbf{D}\boldsymbol{\epsilon}(\mathbf{u})\,\mathrm{d}\Omega = \\ &= \int_0^L \delta\begin{bmatrix}\mathbf{v}\\ \mathbf{v}'\end{bmatrix}^T \int_{\mathscr{S}} \begin{bmatrix}\mathbf{Z}_{\boldsymbol{\xi}}^T\mathbf{D}\mathbf{Z}_{\boldsymbol{\xi}} & \mathbf{Z}_{\boldsymbol{\xi}}^T\mathbf{D}\mathbf{Z}_0\\ \mathbf{Z}_0^T\mathbf{D}\mathbf{Z}_{\boldsymbol{\xi}} & \mathbf{Z}_0^T\mathbf{D}\mathbf{Z}_0\end{bmatrix}\mathrm{d}\mathscr{A}\begin{bmatrix}\mathbf{v}\\ \mathbf{v}'\end{bmatrix}\mathrm{d}x\end{aligned} \tag{5.14}$$

The virtual external work in eq. 5.3 can be discretized as

$$\begin{aligned}\delta\mathscr{L}_e = &\int_0^L \delta\mathbf{v}^T\int_{\mathscr{A}}\mathbf{N}^T\mathbf{f}_\Omega\,\mathrm{d}\mathscr{A}\,\mathrm{d}x + \delta\mathbf{v}^T(L)\int_{\mathscr{S}_L}\mathbf{N}^T\mathbf{f}_L\,\mathrm{d}\mathscr{A} + \\ &+ \int_0^L \delta\mathbf{v}^T\int_{\partial\mathscr{A}}\mathbf{N}^T\boldsymbol{\tau}\,\mathrm{d}\Gamma\,\mathrm{d}x\end{aligned} \tag{5.15}$$

The following matrices can be conveniently defined

$$\begin{aligned}\mathbf{E} &\triangleq \int_{\mathscr{S}}\mathbf{Z}_{\boldsymbol{\xi}}(\boldsymbol{\xi})^T\mathbf{D}\mathbf{Z}_{\boldsymbol{\xi}}(\boldsymbol{\xi})\,\mathrm{d}\mathscr{A}\\ \mathbf{C} &\triangleq \int_{\mathscr{S}}\mathbf{Z}_{\boldsymbol{\xi}}(\boldsymbol{\xi})^T\mathbf{D}\mathbf{Z}_0\,\mathrm{d}\mathscr{A}\\ \mathbf{M} &\triangleq \int_{\mathscr{S}}\mathbf{Z}_0^T\mathbf{D}\mathbf{Z}_0\,\mathrm{d}\mathscr{A}\end{aligned} \tag{5.16}$$

together with the semi-discrete domain and boundary nodal forces[1]

$$\begin{aligned}\mathbf{F}_\Omega &\triangleq \int_{\mathscr{A}} \mathbf{N}^T \mathbf{f}_\Omega \,\mathrm{d}\mathscr{A} \\ \mathbf{F}_L &\triangleq \int_{\mathscr{S}_L} \mathbf{N}^T \mathbf{f}_L \,\mathrm{d}\mathscr{A} \\ \mathbf{F}_{\partial\mathscr{A}} &\triangleq \int_{\partial\mathscr{A}} \mathbf{N}^T \boldsymbol{\tau} \,\mathrm{d}\Gamma\end{aligned} \tag{5.17}$$

so that the virtual work principle (eq. 5.9) can be compactly written in its semi-discrete form as

$$\boxed{\int_0^L \delta \begin{bmatrix}\mathbf{v}\\ \mathbf{v}'\end{bmatrix}^T \begin{bmatrix}\mathbf{E} & \mathbf{C}\\ \mathbf{C}^T & \mathbf{M}\end{bmatrix} \begin{bmatrix}\mathbf{v}\\ \mathbf{v}'\end{bmatrix} \mathrm{d}x = \int_0^L \delta\mathbf{v}^T (\mathbf{F}_\Omega + \mathbf{F}_{\partial\mathscr{A}})\,\mathrm{d}x + \delta\mathbf{v}^T(L)\mathbf{F}_L, \quad \forall\delta\mathbf{v} \in [H_0^1([0,L])]^{n_s\times 3}} \tag{5.18}$$

5.3 Model reduction approaches

In this section, we introduce the idea of projecting the semi-discrete solution vector field

$$\mathbf{v}(x) \in H_0^1([0,L])^{n_s\times 3} \tag{5.19}$$

on a lower-dimensional vector field

$$\mathbf{k}(x) \in H_0^1([0,L])^{n_r} \qquad n_r \ll n_s \times 3 \tag{5.20}$$

by means of a matrix $\mathbf{P} \in \mathbb{R}^{(n_s\times 3)\times n_r}$ so that we can write the projection as

$$\mathbf{v}_\parallel(x) = \mathbf{P}\mathbf{k}(x) \tag{5.21}$$

and use it to replace the full solution $\mathbf{v}(x)$ in eq. 5.18, in accordance to classical methods for over-determined systems [104]. The problem is finding a matrix $\mathbf{P}$ such that the error $\mathbf{v}(x) - \mathbf{v}_\parallel(x)$ is negligible for the application purposes. This projection-based approach is common in many model reduction problems [148, 12]. In this work, we don't look for a mathematical optimality criterion for the definition of matrix $\mathbf{P}$, since this often requires some knowledge of the full-system solution $\mathbf{v}(x)$, like in Proper Orthogonal Decomposition (POD) approaches [12]. Instead, we compare different a priori choices of matrix $\mathbf{P}$ based on an assessment of

[1] Alternatively to the boundary flux $\mathbf{f}_L$, an inhomogeneous boundary condition $\mathbf{v}_L$ could be provided. This case is handled by means of boundary data lifting [138]

$$\mathbf{v}(x) = \mathbf{v}_0(x) + \mathbf{G}(x)\mathbf{v}_L$$

where $\mathbf{v}_0(x)$ is the new unknown function with homogeneous Dirichlet conditions at $x = L$, and $\mathbf{G}(x)$ is an arbitrary matrix function such that $\mathbf{G}(0) = \mathbf{0}$ and $\mathbf{G}(L) = \mathbf{I}$.

their computational cost and convergence trends, thus avoiding full-system computations.

We will investigate two alternative choices for matrix $\mathbf{P}$. Firstly, the eigenfunctions of the strong form of eq. 5.18. Lastly, the singular vectors of the cross-section in-plane deformation energy matrix $\mathbf{E}$.

5.3.1 Eigenvectors of the Hamiltonian system

We now look for a strong form of the semi-discrete variational formulation (eq.5.18). Since variations $\delta\mathbf{v}$ and $\delta\mathbf{v}'$ are not independent, we can integrate by parts eq. 5.18 and impose Dirichlet boundary conditions $\delta\mathbf{v}(0) = \mathbf{0}$, to get

$$\begin{aligned}\int_0^L \delta\mathbf{v}^T \left(\mathbf{E}\mathbf{v} + (\mathbf{C} - \mathbf{C}^T)\mathbf{v}' - \mathbf{M}\mathbf{v}''\right) \mathrm{d}x = \\ = \int_0^L \delta\mathbf{v}^T \left(\mathbf{F}_\Omega + \mathbf{F}_{\partial\mathscr{A}}\right) \mathrm{d}x + \delta\mathbf{v}^T(L)\left(\mathbf{F}_L - \mathbf{C}^T\mathbf{v}(L) - \mathbf{M}\mathbf{v}'(L)\right), \\ \forall\delta\mathbf{v} \in [H_0^1([0,L])]^{n_s\times 3}\end{aligned} \tag{5.22}$$

For a more compact notation, it is convenient to define a new matrix

$$\mathbf{H} \triangleq \mathbf{C} - \mathbf{C}^T \tag{5.23}$$

Imposing the arbitrariness of variations in the domain $\delta\mathbf{v}(x)$ and on the Neumann boundary $\delta\mathbf{v}(L)$ leads us to

$$\begin{aligned}&\mathbf{E}\mathbf{v} + \mathbf{H}\mathbf{v}' - \mathbf{M}\mathbf{v}'' = \mathbf{F}_\Omega + \mathbf{F}_{\partial\mathscr{A}}, \qquad x \in (0, L) \\ &\mathbf{v}(0) = \mathbf{0} \\ &\mathbf{C}^T\mathbf{v}(L) + \mathbf{M}\mathbf{v}'(L) = \mathbf{F}_L\end{aligned} \tag{5.24}$$

Introducing

$$\mathbf{w}(x) \triangleq \mathbf{v}'(x) \tag{5.25}$$

we can write the corresponding homogeneous system of first order differential equations as

$$\underbrace{\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{M}\end{bmatrix}}_{\mathbf{B}} \frac{\mathrm{d}}{\mathrm{d}x} \underbrace{\begin{bmatrix}\mathbf{v}\\ \mathbf{w}\end{bmatrix}}_{\mathbf{q}} = \underbrace{\begin{bmatrix}\mathbf{0} & \mathbf{I}\\ \mathbf{E} & \mathbf{H}\end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix}\mathbf{v}\\ \mathbf{w}\end{bmatrix}}_{\mathbf{q}} \tag{5.26}$$

The above system is Hamiltonian, meaning that the spectrum of matrix pair $(\mathbf{A}, \mathbf{B})$ is symmetric with respect to the imaginary axis. In fact, the same first order system of differential equations can be derived from a Hamiltonian formulation of the semi-discrete problem (eq. 5.18) [125, 18, 19].

Since we have imposed no boundary conditions, the system is twelve times singular, with six rigid modes and six *classical* beam deformation modes (as

shown in [121, 120, 48, 47, 125]). Only four linearly independent eigenvectors, corresponding to the three rigid translations and the rigid rotation around the beam axis, can be computed for the null eigenvalues. So matrix pair $(\mathbf{A},\mathbf{B})$ is defective [74] and it is not diagonalizable. A Jordan form decomposition can be sought [74].

$$\mathbf{A} = \mathbf{BXJX}^{-1} \tag{5.27}$$

where the Jordan matrix $\mathbf{J}$ contains one Jordan block $\mathbf{J}_g$ for each defective eigenpair

$$\mathbf{J}_g = \begin{bmatrix} 0 & 1 & & & & \\ & 0 & 1 & & & \\ & & \ddots & \ddots & & \\ & & & & 0 & 1 \\ & & & & & 0 \end{bmatrix} \in \mathbb{R}^{g\times g} \tag{5.28}$$

where g is the geometric multiplicity of the eigenvector. In this case, the problem is also derogatory [74] since four Jordan blocks are present for eigenvalues with the same value. In the present case, the Jordan blocks are associated to bending (two 4×4 blocks), axial tension (2×2), and torsion (2×2) [125]. The rest of the matrix $\mathbf{J}$ is diagonal and contains the non-null eigenvalues. Thus the Jordan matrix has the following structure [125]

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_4 & & & & & & & \\ & \mathbf{J}_4 & & & & & & \\ & & \mathbf{J}_2 & & & & & \\ & & & \mathbf{J}_2 & & & & \\ & & & & \lambda_{13} & & & \\ & & & & & \lambda_{14} & & \\ & & & & & & \ddots & \\ & & & & & & & \lambda_{n_s\times 6} \end{bmatrix} \tag{5.29}$$

Matrix $\mathbf{X}$ contains the Jordan principal vectors. They coincide with the eigenvectors for non-defective eigenpairs, while in the defective case they are computed sequentially from so-called Jordan chains

$$\begin{aligned} &\mathbf{Aq}_0 = \mathbf{0} \\ &\mathbf{Aq}_i = \mathbf{Bq}_{i-1}, \quad \forall i \geq 1 \end{aligned} \tag{5.30}$$

In the present problem, Jordan chains can be explicitly written [125] as

$$\begin{aligned} &\mathbf{Ed}_0 = \mathbf{0} \\ &\mathbf{Ed}_1 = -\mathbf{Hd}_0 \\ &\mathbf{Ed}_i = \mathbf{Md}_{i-1} - \mathbf{Hd}_{i-2}, \quad \forall i \geq 2 \end{aligned} \tag{5.31}$$

With

$$\mathbf{q}_0 = \begin{bmatrix} \mathbf{d}_0 \\ \mathbf{0} \end{bmatrix}, \qquad \mathbf{q}_i = \begin{bmatrix} \mathbf{d}_i \\ \mathbf{d}_{i-1} \end{bmatrix}, \quad \forall i \geq 1 \tag{5.32}$$

As noted in [74, 125], Jordan blocks correspond to a polynomial solutions. For example, for a generic $m \times m$ block $\mathbf{J}_m(\lambda_k)$ associated to an eigenvalue λ_k

$$e^{\mathbf{J}_m(\lambda_k)x} = e^{\lambda_k x} \begin{bmatrix} 1 & x & \frac{x^2}{2} & \frac{x^3}{3!} & \cdots & \frac{x^{m-1}}{(m-1)!} \\ 0 & 1 & x & \frac{x^2}{2} & \cdots & \frac{x^{m-2}}{(m-2)!} \\ & 0 & 1 & x & \ddots & \vdots \\ & & \ddots & \ddots & \ddots & \vdots \\ & & & \ddots & \ddots & \vdots \\ & & & & 0 & 1 \end{bmatrix} \tag{5.33}$$

Thus, the two 4×4 blocks originate the de Saint Venant's cubic solution for beam bending, while the 2×2 blocks originate the linear axial tension and torsion solutions.

Since the problem is twelve times singular, the eigenvalue computation is ill-conditioned and it is advisable to remove the singularity before performing a numerical eigenvalue computation. The computational procedure is described in appendix A.

Physical interpretation. Eigenvectors related to non-null eigenvalues exhibit a combination of camber-morphing and out-of-plane warping behavior, making them interesting for as a basis functions for morphing applications. Eigenanalysis means decomposing the solution $\mathbf{v}$ on a basis of complex exponentials $e^{\lambda_i x}$, whose amplitudes are determined from boundary conditions. From Euler's formula, these complex exponentials can always be rewritten in trigonometric form, so the eigenvalues real part $\Re(\lambda_i)$ represent the amplification or damping of each modal shape along the x-axis. This motivates the study of modal truncation as a means for building the projection matrix $\mathbf{P}$ for morphing applications, since retaining in the model the eigenvectors with the slowest decay rate would allow to analyze the propagation of an imposed shape along the x-axis.

Assessment of the method. In time evolution problems, eigenfunctions of the first order system appear as a natural choice for basis functions. In fact, the interpretation of the imaginary part of eigenvalues $\Im(\lambda_i)$ as time frequencies is straigthforward. Even if better model reduction approaches could be employed based on the specific analysis needs [148, 12], this physical interpretation is retained in modal truncation of first order linear time evolution problems.

From the computational point of view, some issues when dealing with eigenanalysis of Hamiltonian systems have to be considered.

1. No convergence estimates are available for modal truncation of non symmetric systems, meaning that convergence should be verified in practice by testing with an increasing number of basis functions retained in the reduced model.

2. Eigenvalues of a Hamiltonian system are symmetric with respect to both the real and the imaginary axis. To the numerical practitioner who uses an iterative eigenvalue solver to compute a limited number of eigenvalues, this means that with a classical eigenvalue solver four new eigenvalues have to be computed before an independent new one is found, making the procedure costly unless specific eigenvalue solvers are employed [112, 24, 25].

3. The non symmetric Hamiltonian matrix pair of the problem at hand is ill-conditioned, even after mesh scaling has been employed to improve the condition number of matrices **E**, **C**, **M**, and after deflation procedures [27, 142] have been employed to desingularize the Hamiltonian matrix pair. Ill-conditioning, combined with the high number of eigenvalues which need to be computed due to symmetry, makes it difficult for the eigenvalue solver to converge on a result for a number of eigenvalues sufficient for analysis purposes.

4. Eigenvectors of the Hamiltonian system (eq. 5.26) show no clear uncoupling of out-of-plane warping displacement from in-plane camber-morphing displacement in most of the eigenfunctions, making it even more difficult to determine the number of eigenvalues which need to be retained in the model in order to have a significant reproduction of a camber morphing behaviour, as well as preventing a preliminary study of a purely camber-morphing forcing and response.

5.3.2 Eigenvectors of the in-plane deformation energy

Due to the difficulties with the computation of the eigenfunctions of the linear system (eq. 5.26), we seek an alternative approach for the choice of the basis functions.

The virtual work contribution

$$\delta\mathscr{L}_{\mathbf{E}}(x) \triangleq \delta\mathbf{v}^T(x)\mathbf{E}\mathbf{v}(x) \tag{5.34}$$

contains the work per unit span related to all the in-plane stress components σ_{yy}, σ_{zz}, τ_{yz} plus the in-plane derivatives of the out-of-plane displacement $\frac{\partial u}{\partial y}$, $\frac{\partial u}{\partial z}$. In

fact, matrix $\mathbf{E}$ comes from the splitting of the strain tensor as

$$\boldsymbol{\epsilon} = \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & \partial/\partial y & 0 \\ 0 & 0 & \partial/\partial z \\ \partial/\partial y & 0 & 0 \\ \partial/\partial z & 0 & 0 \\ 0 & \partial/\partial z & \partial/\partial y \end{bmatrix}}_{\triangleq \mathscr{D}_{\boldsymbol{\xi}}} \mathbf{u} + \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}}_{\triangleq \mathbf{S}} \frac{\partial \mathbf{u}}{\partial x} = \mathscr{D}_{\boldsymbol{\xi}} \mathbf{u} + \mathbf{S} \frac{\partial \mathbf{u}}{\partial x} \tag{5.35}$$

and the application of the differential operator $\mathscr{D}_{\boldsymbol{\xi}}$, defined by the components of the strain related only to variations of the 3D displacement along in-plane coordinates, to a discretization matrix $\mathbf{N}(\boldsymbol{\xi})$

$$\mathbf{E} = \int_{\mathscr{A}} \left(\mathscr{D}_{\boldsymbol{\xi}} \mathbf{N}(\boldsymbol{\xi})\right)^T \mathbf{D} \left(\mathscr{D}_{\boldsymbol{\xi}} \mathbf{N}(\boldsymbol{\xi})\right) \mathrm{d}\mathscr{A} \tag{5.36}$$

Since the virtual work in 5.34 vanishes for the three rigid translation and for a rotation about the x axis, matrix $\mathbf{E}$ is positive semidefinite and has four null eigenvalues [125]. Since $\mathbf{E}$ is symmetric by construction, it is diagonalizable, its eigenvalues are real and its eigenvectors are orhogonal.

Defining the cross-section mass matrix

$$\mathbf{W}_{\mathscr{S}} = \int_{\mathscr{A}} \mathbf{N}^T(\boldsymbol{\xi}) \mathbf{N}(\boldsymbol{\xi}) \, \mathrm{d}\mathscr{A} \tag{5.37}$$

the numerical approximation of the most deformable constant shapes is given by the solution of the real symmetric eigenvalue problem

$$\mathbf{E}\mathbf{v}_i = \sigma_i \mathbf{W}_{\mathscr{S}} \mathbf{v}_i \qquad i = 1, \ldots, R \tag{5.38}$$

for the first R lowest-modulus non-null eigenvalues, leading to the definition of a matrix of basis functions

$$\mathbf{P}_E \equiv [\mathbf{v}_1, \ldots, \mathbf{v}_R] \in \mathbb{R}^{3N \times R} \tag{5.39}$$

This matrix is bordered by the de Saint Venant's solution $\mathbf{P}_d$ computed in the previous section, to give the complete projection matrix

$$\mathbf{P} = [\mathbf{P}_E, \mathbf{P}_d] \tag{5.40}$$

Physical interpretation. Looking for the lowest-modulus non-null eigenvalues of matrix pair $(\mathbf{E}, \mathbf{W}_{\mathscr{S}})$ means maximizing the L^2 norm of the solution while considering orthogonal contributions to the internal energy of a deformed shape which doesn't change along the x-axis

$$\arg\min_{\mathbf{v}_i} \left(\mathbf{v}_i^T \mathbf{W}_{\mathscr{S}} \mathbf{v}_i\right) \qquad \text{s.t.} \quad \mathbf{v}_i^T \mathbf{E} \mathbf{v}_i = 1, \quad \mathbf{v}_i \notin \ker(\mathbf{E}) \tag{5.41}$$

For solutions which are constant along the beam axis, this approach reduces to a Proper Orthogonal Decomposition (POD) or Singular Value Decomposition (SVD) [12].

Although an out-of-plane variation of the deformed shape will be present in most of morphing applications, it is often desirable to keep it at a minimum. In a design phase, having a model which uncouples out-of-plane from in-plane shape variations allows to design the cross-section material layout in order to maximize the desired in-plane morphing capabilities.

Assessment of the method. The present projection-based approach offers several practical advantages over the previous one.

1. Clear convergence estimates are available for system SVD (Schmidt-Eckart-Young-Mirsky theorem [12]).

2. Since **E** is symmetric, we know from spectral theory that all its eigenvalues are real and the related eigenfunctions are orthogonal [104],

3. In practice, the previous point means that it is easier for standard eigenvalue solvers to converge on a large number of eigenpairs (in this work, above 300).

4. Furthermore, camber-morphing and warping displacements singular vectors are naturally uncoupled, as it will be shown in the next paragraph.

Uncoupling of *in-plane* and *out-of-plane* displacement. It is now convenient to take a step back, and to analyse the energy associated to the differential operator $\mathscr{D}_{\boldsymbol{\xi}}\mathbf{u}$ before any discretization is introduced

$$\delta\mathscr{L}_{\mathbf{E}}(x) = \int_{\mathscr{A}} \left(\mathscr{D}_{\boldsymbol{\xi}}\delta\mathbf{u}\right)^T \mathbf{D}\left(\mathscr{D}_{\boldsymbol{\xi}}\mathbf{u}\right) \mathrm{d}\mathscr{A} \tag{5.42}$$

If we split the displacement **u** into out-of-plane warping u and in-plane (camber-morphing) displacement **c**

$$\mathbf{u} = \begin{bmatrix} u \\ \mathbf{c} \end{bmatrix} \tag{5.43}$$

we can move the rows of operator $\mathscr{D}_{\boldsymbol{\xi}}$ according to the permutation $[1,4,5,2,3,6]$ so to partition it analogously

$$\mathscr{D}_{\boldsymbol{\xi}} = \left[\begin{array}{c|cc} 0 & 0 & 0 \\ \hline \partial/\partial y & 0 & 0 \\ \partial/\partial z & 0 & 0 \\ \hline 0 & \partial/\partial y & 0 \\ 0 & 0 & \partial/\partial z \\ 0 & \partial/\partial z & \partial/\partial y \end{array}\right] \triangleq \begin{bmatrix} 0 & \mathbf{0}^{1\times 2} \\ \mathscr{D}^{w}_{\boldsymbol{\xi}} & \mathbf{0}^{2\times 2} \\ \mathbf{0}^{3\times 1} & \mathscr{D}^{c}_{\boldsymbol{\xi}} \end{bmatrix} \tag{5.44}$$

Thus, its terms act diagonally on u or $\mathbf{c}$

$$\mathscr{D}_{\boldsymbol{\xi}}\mathbf{u} = \begin{bmatrix} 0 & \mathbf{0}^{1\times 2} \\ \mathscr{D}^{w}_{\boldsymbol{\xi}} & \mathbf{0}^{2\times 2} \\ \mathbf{0}^{3\times 1} & \mathscr{D}^{c}_{\boldsymbol{\xi}} \end{bmatrix} \begin{bmatrix} u \\ \mathbf{c} \end{bmatrix} \tag{5.45}$$

By introducing the same partitioning of the elasticity matrix $\mathbf{D}$ (eq. 5.8) through the application of the same permutation $[1,4,5,2,3,6]$ to its rows and columns

$$\mathbf{D} = \left[\begin{array}{c|cc|cc} 2\mu+\lambda & 0 & 0 & \lambda & \lambda & 0 \\ \hline 0 & \mu & 0 & 0 & 0 & 0 \\ 0 & 0 & \mu & 0 & 0 & 0 \\ \hline \lambda & 0 & 0 & 2\mu+\lambda & \lambda & 0 \\ \lambda & 0 & 0 & \lambda & 2\mu+\lambda & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu \end{array}\right] = \begin{bmatrix} \mathbf{D}_{00} & \mathbf{0}^{1\times 2} & \mathbf{D}_{0w} \\ \mathbf{0}^{2\times 1} & \mathbf{D}_{ww} & \mathbf{0}^{2\times 3} \\ \mathbf{D}^{T}_{0w} & \mathbf{0}^{3\times 2} & \mathbf{D}_{cc} \end{bmatrix} \tag{5.46}$$

inserting this partitioning into the expression for the in-plane elastic energy and using the adjoint operators $\mathscr{D}^{w^\dagger}_{\boldsymbol{\xi}}$ and $\mathscr{D}^{c^\dagger}_{\boldsymbol{\xi}}$ we get

$$\begin{aligned} &\delta\mathscr{L}_{\mathbf{E}}(x) = \\ &= \int_{\mathscr{A}} \left(\begin{bmatrix} 0 & \mathbf{0}^{1\times 2} \\ \mathscr{D}^{w}_{\boldsymbol{\xi}} & \mathbf{0}^{2\times 2} \\ \mathbf{0}^{3\times 1} & \mathscr{D}^{c}_{\boldsymbol{\xi}} \end{bmatrix} \begin{bmatrix} \delta u \\ \delta\mathbf{c} \end{bmatrix} \right)^{T} \begin{bmatrix} \mathbf{D}_{00} & \mathbf{0}^{1\times 2} & \mathbf{D}_{0w} \\ \mathbf{0}^{2\times 1} & \mathbf{D}_{ww} & \mathbf{0}^{2\times 3} \\ \mathbf{D}^{T}_{0w} & \mathbf{0}^{3\times 2} & \mathbf{D}_{cc} \end{bmatrix} \begin{bmatrix} 0 & \mathbf{0}^{1\times 2} \\ \mathscr{D}^{w}_{\boldsymbol{\xi}} & \mathbf{0}^{2\times 2} \\ \mathbf{0}^{3\times 1} & \mathscr{D}^{c}_{\boldsymbol{\xi}} \end{bmatrix} \begin{bmatrix} u \\ \mathbf{c} \end{bmatrix} \mathrm{d}\mathscr{A} = \\ &= \int_{\mathscr{A}} \begin{bmatrix} \delta u \\ \delta\mathbf{c} \end{bmatrix}^{T} \begin{bmatrix} \mathscr{D}^{w^\dagger}_{\boldsymbol{\xi}}\mathbf{D}_{ww}\mathscr{D}^{w}_{\boldsymbol{\xi}} & \mathbf{0}^{1\times 2} \\ \mathbf{0}^{2\times 1} & \mathscr{D}^{c^\dagger}_{\boldsymbol{\xi}}\mathbf{D}_{cc}\mathscr{D}^{c}_{\boldsymbol{\xi}} \end{bmatrix} \begin{bmatrix} u \\ \mathbf{c} \end{bmatrix} \mathrm{d}\mathscr{A} \end{aligned} \tag{5.47}$$

Thus matrix $\mathbf{E}$ appears to be the discretization of a differential operator which uncouples out-of-plane from in-plane displacements.

5.3.3 Orthogonalization

Although energy modes are already orthonormal, classical deformation modes are not, neither with respect to the energy modes, nor among themselves. For this

reason, a new, orthonormal basis $\tilde{\mathbf{P}}$ in the $L^2(\mathscr{S})$ inner product is obtained from the old one $\bar{\mathbf{P}}$ by means of a Gram-Schmidt process

$$\bar{\mathbf{P}} = \tilde{\mathbf{P}}\mathbf{T} \tag{5.48}$$

where $\mathbf{T}$ is a non-singular upper-triangular $(R+6)\times(R+6)$ matrix. Orthonormality in the $L^2(\mathscr{S})$ inner product is obtained by requiring the orthonormality with respect to the cross-section mass matrix $\mathbf{W}_{\mathscr{S}}$

$$\tilde{\mathbf{P}}^T\mathbf{W}_{\mathscr{S}}\tilde{\mathbf{P}} = \mathbf{I} \tag{5.49}$$

5.4 Fully discrete formulation of the reduced-order model

The finite element discretization of the axis direction outlined in this section allows us to derive the final algebraic system of equations to be solved for the three-dimensional simulation. This procedure is independent from the specific choice of the basis functions used to define the projection matrix $\mathbf{P}$.

5.4.1 Model projection

The aim is to find a projection matrix $\mathbf{P} \in \mathbb{R}^{(n_s\times 3)\times n_r}$ such that its column vectors $\{\mathbf{p}_r\}_{r=1,\dots,n_r}$ are a subset of a vector basis for $\mathbb{R}^{n_s\times 3}$, with $n_r \ll n_s\times 3$. The subspace projection reads

$$\begin{aligned}\mathbf{v}(x) &= \mathbf{P}\mathbf{k}(x)\\ \delta\mathbf{v}(x) &= \mathbf{P}\delta\mathbf{k}(x)\end{aligned} \tag{5.50}$$

both for the unknown solution $\mathbf{v}$ and the test functions $\delta\mathbf{v}$.

Using the above relations in the semi-discrete form of the virtual internal work (eq. 5.18) leads to the definition of the following projected matrices

$$\begin{aligned}\mathbf{E}^\pi &= \mathbf{P}^T\mathbf{E}\mathbf{P}\\ \mathbf{C}^\pi &= \mathbf{P}^T\mathbf{C}\mathbf{P}\\ \mathbf{M}^\pi &= \mathbf{P}^T\mathbf{M}\mathbf{P}\end{aligned} \tag{5.51}$$

So the virtual internal work finally reads

$$\delta\mathscr{L}_i^\pi = \int_0^L \begin{bmatrix}\mathbf{k}\\ \mathbf{k}'\end{bmatrix}^T \begin{bmatrix}\mathbf{E}^\pi & \mathbf{C}^\pi\\ \mathbf{C}^{\pi^T} & \mathbf{M}^\pi\end{bmatrix} \begin{bmatrix}\mathbf{k}\\ \mathbf{k}'\end{bmatrix} \mathrm{d}x \tag{5.52}$$

This expression is independent from the specific choice of the basis functions. Analogously, the projection of the forcing terms can be defined as

$$\begin{aligned}\mathbf{F}_\Omega^\pi &= \mathbf{P}^T\mathbf{F}_\Omega\\ \mathbf{F}_{\partial\mathscr{A}}^\pi &= \mathbf{P}^T\mathbf{F}_{\partial\mathscr{A}}\\ \mathbf{F}_L^\pi &= \mathbf{P}^T\mathbf{F}_L\end{aligned} \tag{5.53}$$

So that the projection of the virtual external work is expressed as

$$\delta \mathscr{L}_e^{\pi} = \int_0^L \delta \mathbf{k}^T \left(\mathbf{F}_{\Omega}^{\pi} + \mathbf{F}_{\partial \mathscr{A}}^{\pi} \right) \mathrm{d}x + \delta \mathbf{k}^T(L) \mathbf{F}_L^{\pi} \tag{5.54}$$

5.4.2 Beam axis discretization and fully discrete formulation

We now introduce a set of nodal basis functions along the x-axis, so that the unknown is discretized as

$$k^{i_r}(x) = \sum_{i_x=1}^{N_x+1} \theta_{i_x}(x) h_{i_x}, \qquad i_r = 1, ..., N_r \tag{5.55}$$

or, in matrix form

$$\mathbf{k}(x) = \boldsymbol{\Theta}(x) \mathbf{h} \tag{5.56}$$

It is now possible to define the linear system matrix

$$\mathbf{L} = \int_0^L \left(\boldsymbol{\Theta}^T \mathbf{E}^{\pi} \boldsymbol{\Theta} + \boldsymbol{\Theta}^T \mathbf{C}^{\pi} \frac{\mathrm{d}\boldsymbol{\Theta}}{\mathrm{d}x} + \frac{\mathrm{d}\boldsymbol{\Theta}}{\mathrm{d}x}^T \mathbf{C}^{\pi^T} \boldsymbol{\Theta} + \frac{\mathrm{d}\boldsymbol{\Theta}}{\mathrm{d}x}^T \mathbf{M}^{\pi} \frac{\mathrm{d}\boldsymbol{\Theta}}{\mathrm{d}x} \right) \mathrm{d}x \tag{5.57}$$

and the forcing array

$$\mathbf{b} = \int_0^L \boldsymbol{\Theta}^T \left(\mathbf{F}_{\Omega}^{\pi} + \mathbf{F}_{\partial \mathscr{A}}^{\pi} \right) \mathrm{d}x + \boldsymbol{\theta}^T(L) \mathbf{F}_L^{\pi} \tag{5.58}$$

so that the final expression of the algebraic linear system reads

$$\mathbf{L}\mathbf{h} = \mathbf{b} \tag{5.59}$$

5.5 Nondimensionalization

Through dimensional analysis, the physical dimensions of the matrices involved in the solution are related to the dimension of a pressure p and a length l in the following way

$$\begin{aligned} \mathbf{E} &\sim [p] \\ \mathbf{C} &\sim [p][l] \\ \mathbf{M} &\sim [p][l]^2 \\ \mathbf{R} &\sim [p][l] \\ \mathbf{L} &\sim [p][l]^2 \\ \mathbf{W}_{\mathscr{S}} &\sim [l]^2 \end{aligned} \tag{5.60}$$

Defining a reference elastic modulus $\mathscr{P}$ and a reference length $\mathscr{L}$, after the assembly phase on the original grid all quantities are nondimensionalized in the following way

$$x \to \mathscr{L} x \qquad \mathrm{d}x \to \mathscr{L}\, \mathrm{d}x \qquad \frac{\partial}{\partial x} \to \frac{1}{\mathscr{L}} \frac{\partial}{\partial x} \qquad \mathbf{u} \to \mathscr{L} \mathbf{u} \tag{5.61}$$

$$\mathbf{E} \to \frac{\mathbf{E}}{\mathscr{P}} \qquad \mathbf{C} \to \frac{\mathbf{C}}{\mathscr{P}\mathscr{L}} \qquad \mathbf{M} \to \frac{\mathbf{M}}{\mathscr{P}\mathscr{L}^2} \qquad \mathbf{W}_{\mathscr{S}} \to \frac{\mathbf{W}_{\mathscr{S}}}{\mathscr{L}^2} \tag{5.62}$$

Nondimensionalization of the gravity forcing, considering the gravity value g and a reference material density ρ, leads to the definition of the following nondimensional parameter

$$\frac{\rho g \mathscr{L}}{\mathscr{P}} \tag{5.63}$$

while nondimensionalization of an aerodynamic load with chord c and asymptotic dynamic pressure q_∞ leads to the definition of the nondimensional parameter

$$\frac{q_\infty c^3}{\mathscr{P}\mathscr{L}^2} \tag{5.64}$$

5.6 Validation

In [125], validation of the projected model is done based on the convergence of beam stiffness matrix terms. Here, we propose a simple validation of the three-dimensional integration of the projected model, based on comparison with classical tension–bending beam theory. We consider a cantilever prismatic solid, loaded at the free end, so that the three-dimensional propagation of extremity effects from the clamped root provides allows to verify the convergence of the projection towards the full model.

This is done for a thin-walled box beam of outer dimensions $L_y = 1, L_y = 0.5$ and thickness $t = 0.05$, made of homogeneous elastic material, whose center of mass coincides with the origin of the reference frame. Formulas for the end-displacement from classical beam theory are used to calculate the end load necessary to achieve a desired displacement. Then, the end load is converted into a distributed pressure/shear load

$$\mathbf{f}_L = \begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} \tag{5.65}$$

to be applied on the end section. Table 5.1 shows the relevant formulas for torque end loads M_y, M_z and a gravity loading. Cross-section area moments are defined as

$$\begin{aligned} S_y &= \int_{\mathscr{A}} z \,\mathrm{d}\mathscr{A}, \qquad S_z &&= \int_{\mathscr{A}} y \,\mathrm{d}\mathscr{A} \\ I_{yy} &= \int_{\mathscr{A}} z^2 \,\mathrm{d}\mathscr{A}, \qquad I_{zz} &&= \int_{\mathscr{A}} y^2 \,\mathrm{d}\mathscr{A} \end{aligned} \tag{5.66}$$

Table 5.2 shows the percent error $e_{\%}$ in the mean end-section displacement component u_j in the generic direction j with respect to the reference value u_j^{ref}

computed from beam theory

$$e_{\%} = \frac{u_j - u_j^{\text{ref}}}{u_j^{\text{ref}}} \times 100 \tag{5.67}$$

for increasing number of intervals N_x along the x-axis, and increasing number of **E** eigenvectors N_r. Classical deformations $\mathbf{P}_d$ are always retained in the model. Figure 5.2 shows the magnified displacement of the three-dimensional prism.

Loading	Displacement	Inverse load	Surface load
M_y	$w_L = -\frac{L^2}{2EI_{yy}} M_y$	$M_y = -\frac{2EI_{yy}}{L^2} v_L$	$f_x = \left(z - \frac{S_y}{A}\right) \frac{M_y}{I_{yy} - \frac{S_y^2}{A}}$
M_z	$v_L = \frac{L^2}{2EI_{zz}} M_z$	$M_z = \frac{2EI_{zz}}{L^2} w_L$	$f_x = \left(-y + \frac{S_z}{A}\right) \frac{M_z}{I_{zz} - \frac{S_z^2}{A}}$
g	$w_L = -\frac{L^4}{8EI_{yy}} \rho g A$	–	–

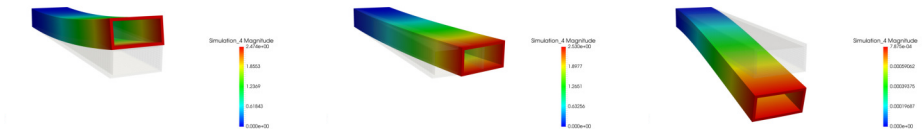
Table 5.1 – End displacement and loading from classical beam theory for selected conditions.

5.7 Analysis of camber-morphing wing sections

We analyze the FishBAC (*fishbone active camber*) cross-section proposed by Woods and Friswell in [173] for camber-morphing applications, and further studied in successive works ([175, 174, 172, 171, 62] among others). This is a thin-

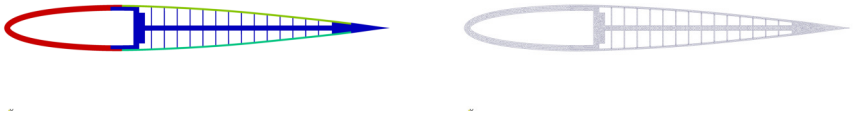
Load, N_x	$N_r = 0$	$N_r = 10$	$N_r = 40$	$N_r = 80$
$M_y, N_x = 25$	-4.8793	-4.6934	-4.3643	-4.3793
$M_y, N_x = 50$	-2.4906	-2.2489	-1.6885	-1.6901
$M_y, N_x = 100$	-1.8709	-1.6118	-0.9239	-0.9117
$M_z, N_x = 25$	-2.2939	-2.1989	-2.0884	-2.0830
$M_z, N_x = 50$	-1.3098	-1.1811	-0.9601	-0.9428
$M_z, N_x = 100$	-1.0504	-0.9098	-0.6070	-0.5769
$g, N_x = 25$	-0.2496	0.4460	1.3741	1.3881
$g, N_x = 50$	2.3526	3.1453	4.5128	4.5391
$g, N_x = 100$	3.0305	3.8562	5.4686	5.5157

Table 5.2 – Percent error in the computed mean end-displacement (eq. 5.67), with varying number of intervals N_x along the x-axis, and varying number of **E** eigenvectors N_r, for selected load conditions.



a M_y, 0.2×. b M_z, 0.2×. c Gravity, 1000×.

Figure 5.2 – Box beam displacement for different loading conditions.



a FishBAC airfoil materials distribution. b FishBAC airfoil mesh.

Figure 5.3 – FishBAC airfoil materials and mesh.

walled cross-section layout built inside a NACA 0012 airfoil. A sketch of the configuration, with materials layout and the mesh used throughout the analysis is given in fig. 5.3. A unit aerodynamic chord is used in the simulations.

5.7.1 Eigenanalysis

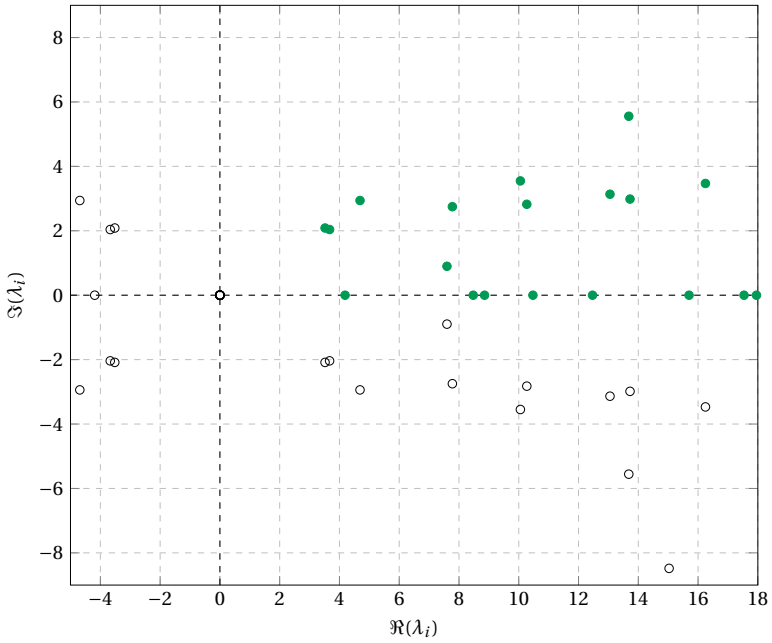
A comparison of the Hamiltonian system and the **E** matrix eigenvalues is shown in figures 5.4.

Eigenfunctions related to matrix **E** are shown in figures 5.5, 5.6. Rigid modes (mode 1 to 4) are omitted. Coloring represents the out-of-plane displacement; from the colorbar, it can be seen that in-plane and out-of-plane displacements are uncoupled in the limits of numerical approximation. Figure 5.7 shows some purely *warping* eigenfunctions, now tilted in order to show the out-of-plane displacement.

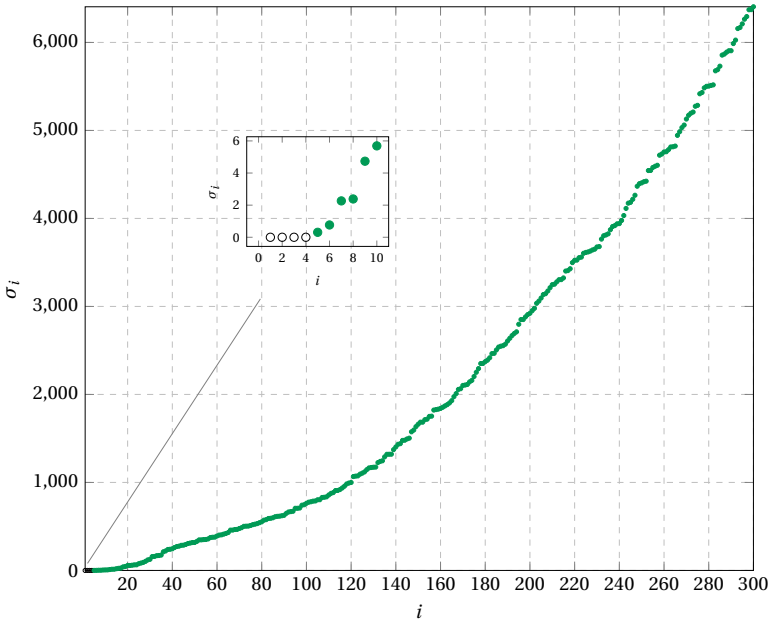
5.7.2 Three-dimensional simulations with gravity loading

We study a gravity load on the extrusion of length $L = 5$ of the FishBAC cross-section. This is done by imposing a volume forcing

$$\mathbf{f}_\Omega = \rho g \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} \tag{5.68}$$

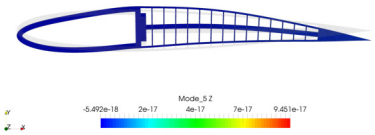


a Eigenvalues of the Hamiltonian system.

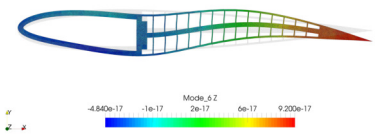


b Eigenvalues of matrix pair $(\mathbf{E}, \mathbf{W}_{\mathscr{S}})$.

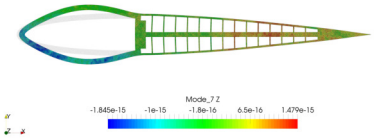
Figure 5.4 – Eigenvalues of the Hamiltonian system and matrix pair $(\mathbf{E}, \mathbf{W}_{\mathscr{S}})$ for the Fish-BAC airfoil.



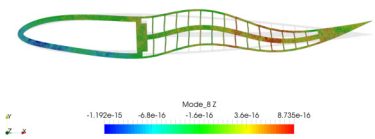
a Eigenfunction 5.



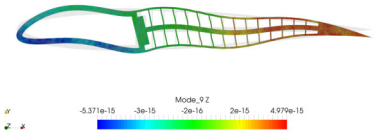
b Eigenfunction 6.



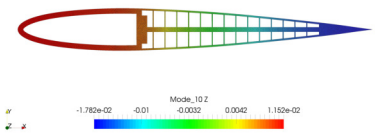
c Eigenfunction 7.



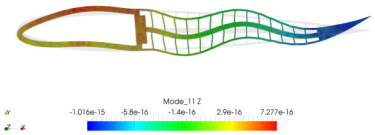
d Eigenfunction 8.



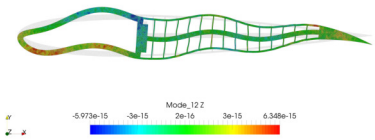
e Eigenfunction 9.



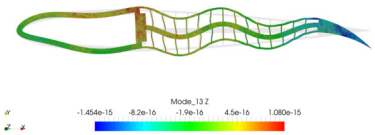
f Eigenfunction 10.



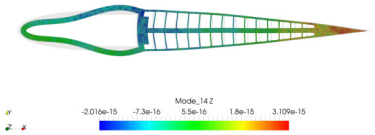
g Eigenfunction 11.



h Eigenfunction 12.

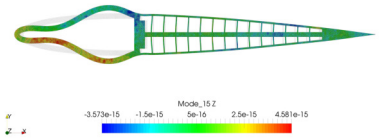


i Eigenfunction 13.

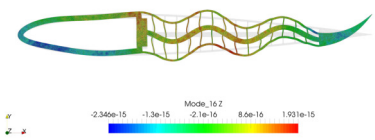


j Eigenfunction 14.

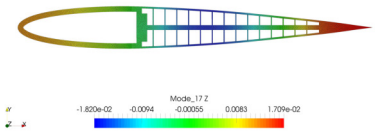
Figure 5.5 – Eigenfunctions from matrix pair $(\mathbf{E}, \mathbf{W}_{\mathscr{S}})$ for the FishBAC airfoil (rescaled with unit $L^2(\mathscr{S})$ norm).



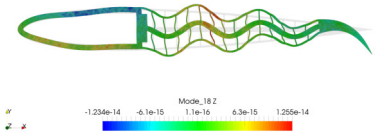
a Eigenfunction 15.



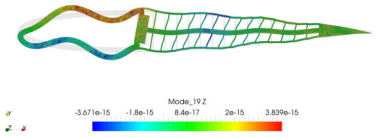
b Eigenfunction 16.



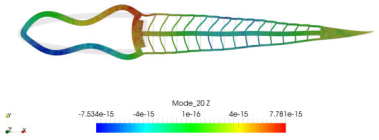
c Eigenfunction 17.



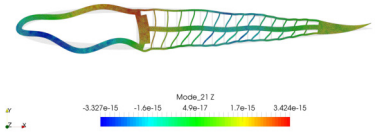
d Eigenfunction 18.



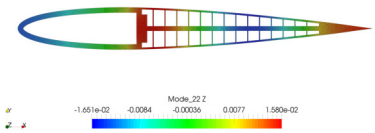
e Eigenfunction 19.



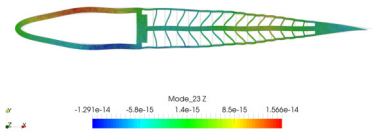
f Eigenfunction 20.



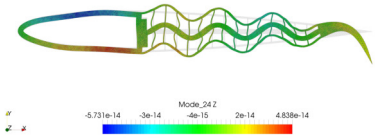
g Eigenfunction 21.



h Eigenfunction 22.

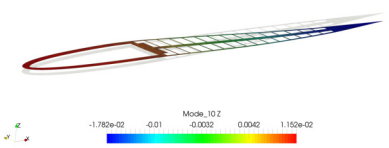


i Eigenfunction 23.

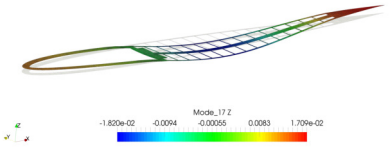


j Eigenfunction 24.

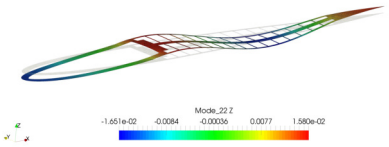
Figure 5.6 – Eigenfunctions from matrix pair $(\mathbf{E}, \mathbf{W}_{\mathscr{S}})$ for the FishBAC airfoil (rescaled with unit $L^2(\mathscr{S})$ norm) – Continued.



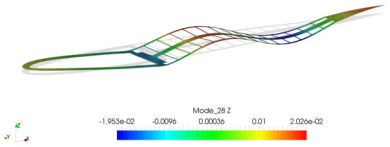
a Eigenfunction 10.



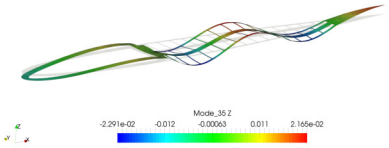
b Eigenfunction 17.



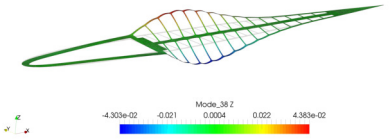
c Eigenfunction 22.



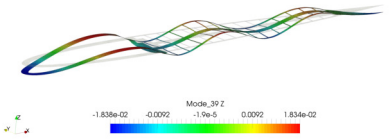
d Eigenfunction 28.



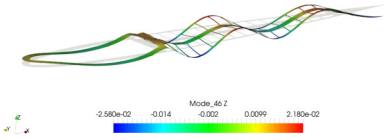
e Eigenfunction 35.



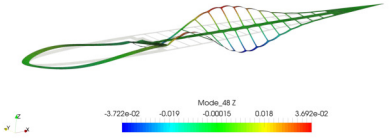
f Eigenfunction 38.



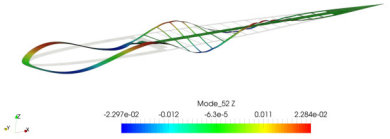
g Eigenfunction 39.



h Eigenfunction 46.



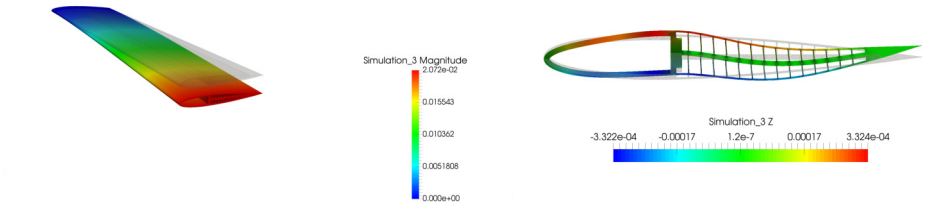
i Eigenfunction 48.



j Eigenfunction 52.

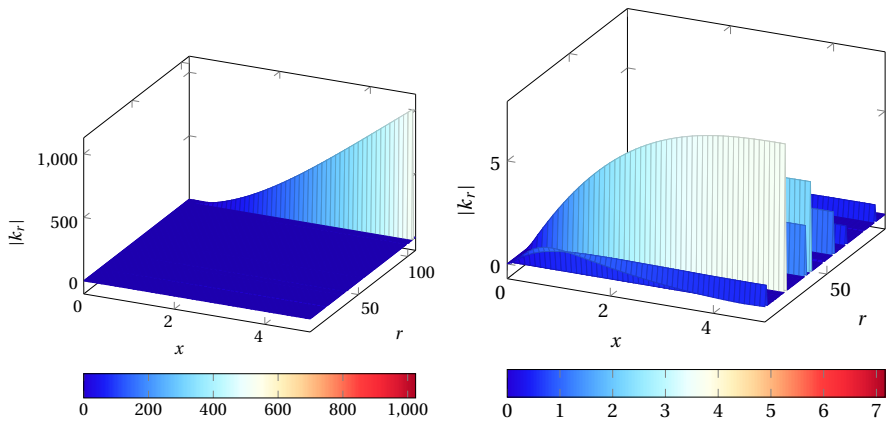
Figure 5.7 – Eigenfunctions from matrix pair $(\mathbf{E}, \mathbf{W}_{\mathscr{S}})$ related to *out–of–plane warping* (rescaled with unit $L^2(\mathscr{S})$ norm).

which is then nondimensionalized as outlined in section 5.5. Figure 5.8 shows the displacement of the wing. A significant camber-morphing behaviour is not directly appreciable from fig. 5.8a, but the modal decomposition allows to filter-out the displacement on the classical deformations $\mathbf{P}_d$, so to better visualize the slight morphing deformation which the model is able to capture (fig.5.8b). This is supported by fig. 5.9, which shows the solution components $\mathbf{k}(x)$ varying along the x-axis. Classical deformation are the last components, overshadowing all other contributions. Figure 5.9b shows only components related to non-classical shapes $\mathbf{P}_\mathbf{E}$, showing that some camber-morphing effect in fact develops along the wing axis.



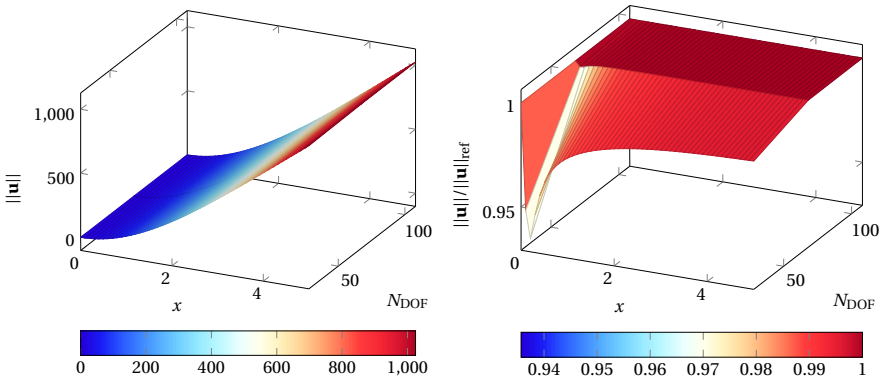
a Wing displacement, 10×. b End-section (only morphing), 500×.

Figure 5.8 – Displacement due to gravity on the FishBAC wing.



a Solution components $\mathbf{k}(x)$. b Solution components $\mathbf{k}(x)$, classical deformation not shown.

Figure 5.9 – Solution components $\mathbf{k}(x)$ for the gravity load.



a Absolute $L^2(\mathscr{S})$ norm, varying DOFs. b Relative $L^2(\mathscr{S})$ norm, varying DOFs.

Figure 5.10 – Displacement convergence with gravity load.

5.7.3 Three-dimensional simulation on imposed modal shapes

From the definition of the eigenvectors of matrix pair $(\mathbf{E}, \mathbf{W}_{\mathscr{S}})$, it is possible to design a forcing leading to a constant morphing solution on a mode $\mathbf{v}_i$ along the x-axis by simply choosing the forcing terms as

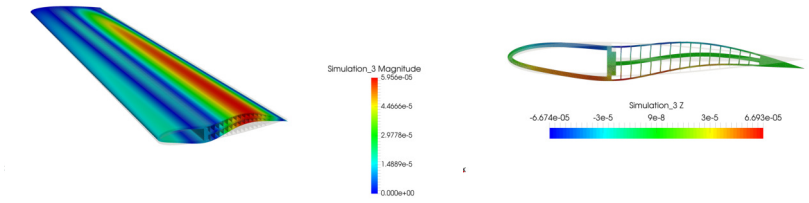
$$\begin{aligned} \mathbf{F}_\Omega &= \mathbf{W}_{\mathscr{S}}\mathbf{v}_i \\ \mathbf{F}_L &= \mathbf{C}^T\mathbf{v}_i \end{aligned} \tag{5.69}$$

and setting an inhomogeneous Dirichlet boundary condition at the root on the desired mode $\mathbf{v}(0) = \mathbf{v}_i$. The same could be done for any prescribed solution made of a desired linear combination of eigenvectors.

In order to study the evolution of the morphing shape along the x-axis, it is more interesting to impose the same forcing while leaving the root boundary conditions unchanged, i.e. on the undeformed configuration. This is done separately for eigenvectors 6 and 8 (previously shown in fig. 5.5). Results for displacement and solution components for forcing on eigenvector 6 are shown in figures 5.11, 5.12, while for eigenvector 8 they are shown in figures 5.14, 5.15. A moderate redistribution of energy on classical deformation is present, A convergence study for the $L^2(\mathscr{S})$ norm of the displacement is shown in figures 5.13, 5.16, varing the number of modes retained in the model. Relative variation of the $L^2(\mathscr{S})$ norm with respect with the final simulation on 100 modes is shown in figures 5.13b, 5.16b.

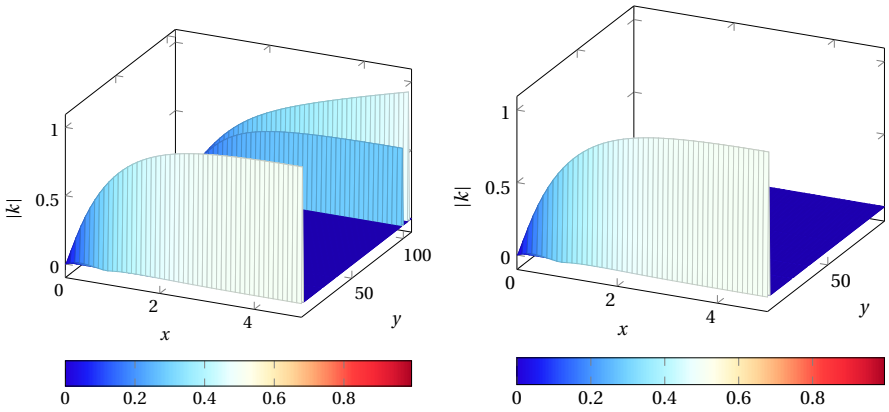
5.7.4 Three-dimensional simulations with analytical pressure loading

A preliminary three-dimensional simulation with surface loading is performed by assigning an analytical pressure distribution on the lateral surface $\partial\Omega\backslash\{\mathscr{S}_0 \cup \mathscr{S}_L\}$.



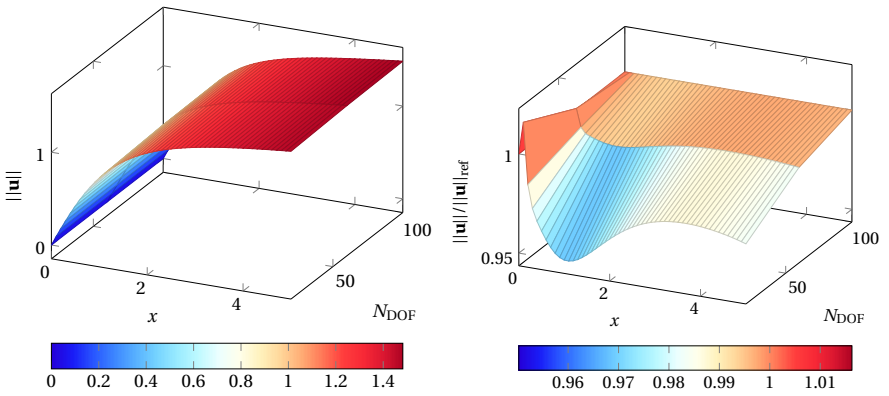
a Wing displacement, 1000×. b End-section, 1000×.

Figure 5.11 – Displacement due to forcing on eigenvector 6 for the FishBAC airfoil.



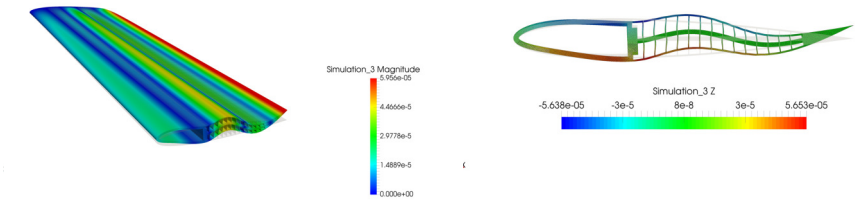
a Solution components $\mathbf{k}(x)$. b Solution components $\mathbf{k}(x)$, classical deformations not shown.

Figure 5.12 – Solution components $\mathbf{k}(x)$ for imposed forcing on eigenvector 6.



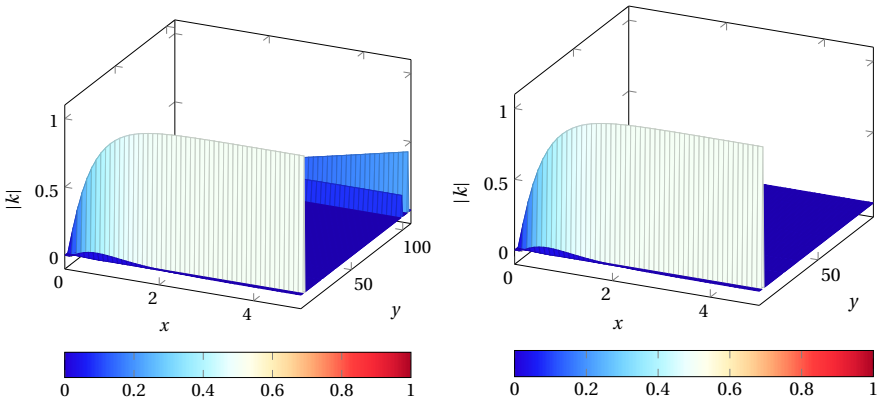
a Absolute $L^2(\mathscr{S})$ norm, varying DOFs. b Relative $L^2(\mathscr{S})$ norm, varying DOFs.

Figure 5.13 – Convergence study for imposed forcing on eigenvector 6.



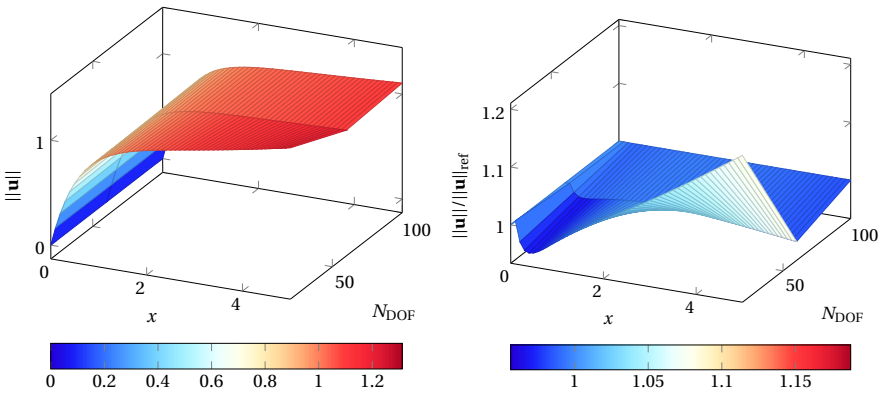
a Mode 6, 1000×. b Mode 8, 1000×.

Figure 5.14 – Displacement due to forcing on eigenvector 8 for the FishBAC airfoil.



a All modes. b All modes.

Figure 5.15 – Solution components $\mathbf{k}(x)$ for imposed forcing on eigenvector 8.



a Absolute $L^2(\mathscr{S})$ norm, varying DOFs. b Relative $L^2(\mathscr{S})$ norm, varying DOFs.

Figure 5.16 – Convergence study for imposed forcing on eigenvector 8.

Defining the coordinate

$$\theta(\boldsymbol{\xi}) = \text{sign}(z)\arccos\left(2\frac{y-y_0}{y_{\max}-y_{\min}}-1\right) \tag{5.70}$$

an analytical pressure function is defined as

$$P(x,\boldsymbol{\xi}) = \frac{1}{L}\sqrt{L^2-x^2}\left(C+A_0\sin^2\theta(\boldsymbol{\xi})+A_1\sin\theta(\boldsymbol{\xi})\right) \tag{5.71}$$

Parameter values $C=1$, $A_0=0.5$, $A_1=0.5$ have been used in the simulations. The rationale is to assign a chordwise pressure distribution composed of a constant part plus a symmetrical contribution ($\sin^2\theta$) and an asymmetrical one ($\sin^2\theta$). This chordwise pressure distribution is modulated by an elliptical distribution along the x-axis. Finally, resultants are filtered out by defining the surface load $\boldsymbol{\tau}$ as

$$\boldsymbol{\tau}(x,\boldsymbol{\xi}) = P(x,\boldsymbol{\xi})\mathbf{n}(\boldsymbol{\xi}) - \int_{\partial\mathscr{A}} P(x,\boldsymbol{\xi})\mathbf{n}(\boldsymbol{\xi})\,\mathrm{d}\Gamma \tag{5.72}$$

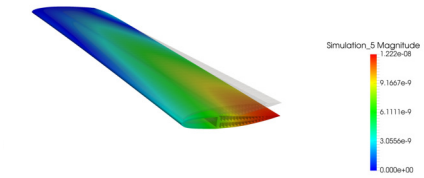
Two materials layouts are studied; they are summarized in table 5.3. Different scalings for the elastic modulus of different materials (identified with reference to their color in figure 5.3) are chosen in order to enhance the camber-morphing capabilities of the airfoil (with panels stiffness varying from 0.25 in configuration A to 0.05 in configuration B) while keeping a good stiffness in the airfoil nose. Three-dimensional and cross-section displacements are compared in fig. 5.17. The displacement is distributed on various modes whose amplitude do vary along the x-axis. Configuration B exhibits a remarked variation along the wing axis. Solution components for both configurations are shown in fig. 5.18, while displacements convergence in the $L^2(\mathscr{S})$ norm is shown in fig. 5.17.

	Red	Blue	Yellow	Green
Configuration A	4	0.5	0.25	0.25
Configuration B	4	0.5	0.05	0.05

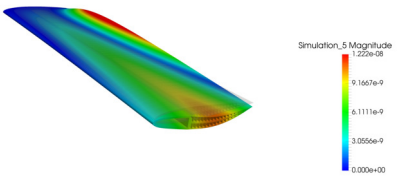
Table 5.3 – Elastic modulus scaling for each cross-section material (with reference to fig. 5.3).

5.8 Fluid–structure interface

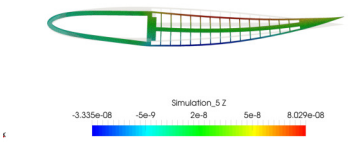
In order to be able to perform a computational aeroelasticity simulation, a coupling between a structural mechanics solver and a computational fluid dynamics solver is needed. In general, different coupled solution strategies and different load transfer algorithms can be conceived [79].



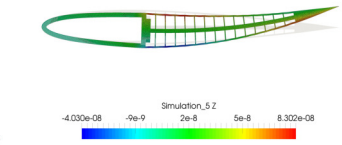
a Wing displacement, configuration A, $10^7\times$.



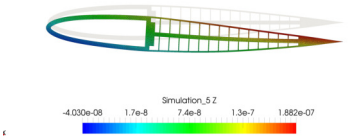
b Wing displacement, configuration B, $10^7\times$.



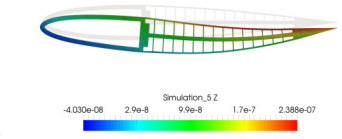
c Section at $x = 1$, configuration A, $10^7\times$.



d Section at $x = 1$, configuration B, $10^7\times$.

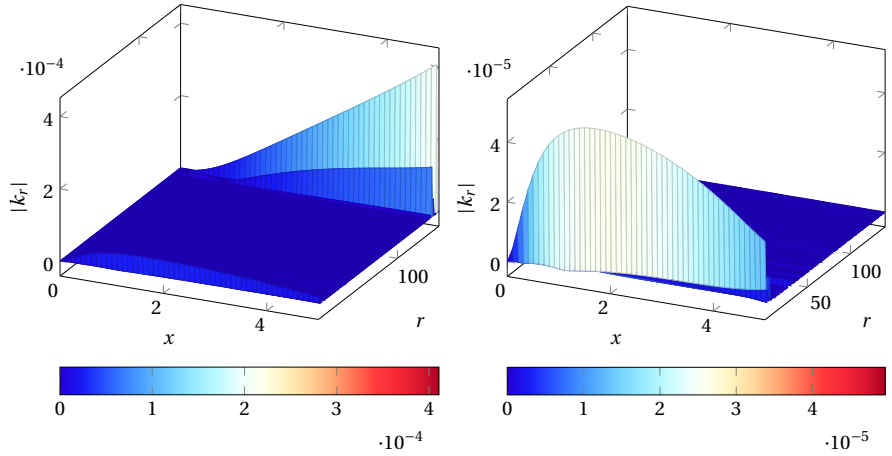


e End-section, configuration A, $10^7\times$.



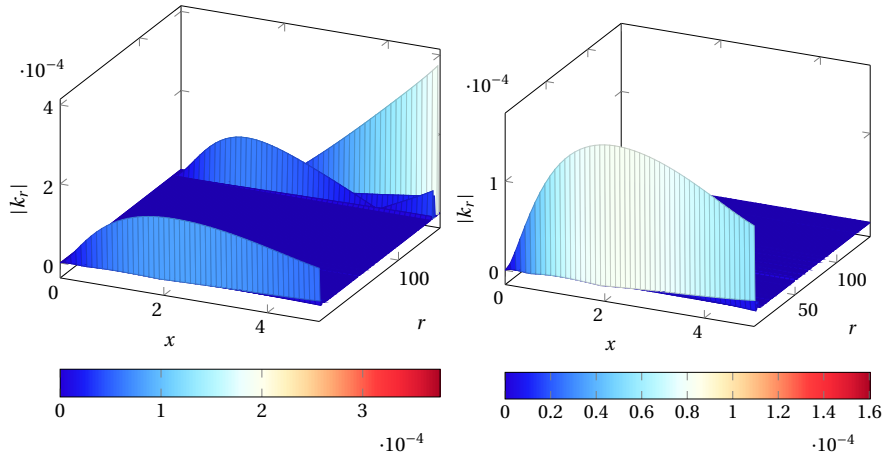
f End-section, configuration B, $10^7\times$.

Figure 5.17 – Displacement due to analytical pressure load, for configuration A (left) and B (right).



a Solution components $\mathbf{k}(x)$, configuration A.

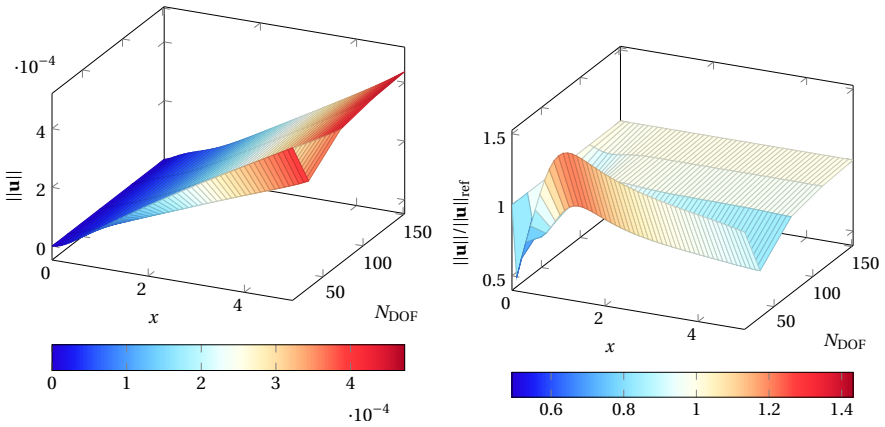
b Solution components $\mathbf{k}(x)$ (classical deformations not shown), configuration A.



c Solution components $\mathbf{k}(x)$, configuration B.

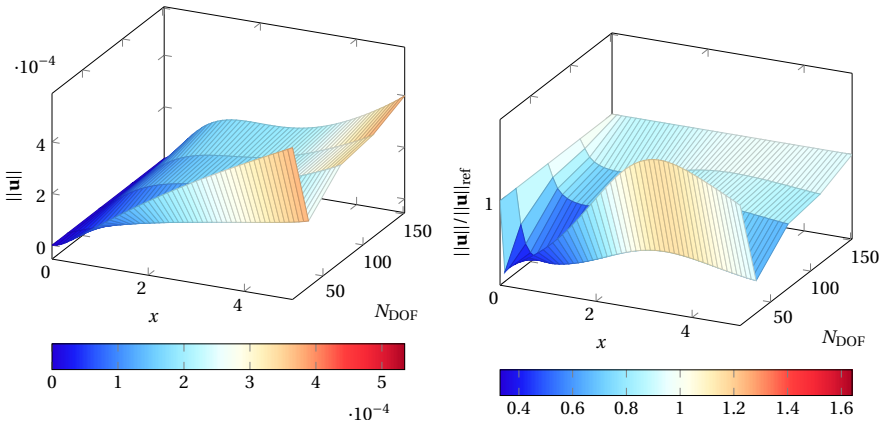
d Solution components $\mathbf{k}(x)$ (classical deformations not shown), configuration B.

Figure 5.18 – Solution components $\mathbf{k}(x)$ with analytical pressure load, for configuration A (top) and B (bottom).



a Absolute $L^2(\mathscr{S})$ norm, varying DOFs, configuration A.

b Relative $L^2(\mathscr{S})$ norm, varying DOFs, configuration A.



c Absolute $L^2(\mathscr{S})$ norm, varying DOFs, configuration B.

d Relative $L^2(\mathscr{S})$ norm, varying DOFs, configuration B.

Figure 5.19 – Displacement convergence with analytical pressure load, for configuration A (top) and B (bottom).

A fundamental issue arises when independent meshes are employed in the solid and fluid domains, so that non-matching discretizations are used on the solid and fluid side of the same aerodynamic surface. Since interpolation deals with discrete nodal values of $\mathbf{u}(\mathbf{x})$ and $\mathbf{f}(\mathbf{x})$, it is convenient to introduce the discretizations[2]

$$\begin{aligned} \mathbf{u}^{(F)}(\mathbf{x}) &= \boldsymbol{\phi}^{(F)}(\mathbf{x})\mathbf{u}^{(F)} \\ \mathbf{u}^{(S)}(\mathbf{x}) &= \boldsymbol{\phi}^{(S)}(\mathbf{x})\mathbf{u}^{(S)} \end{aligned} \tag{5.73}$$

In this case, the displacement computed on the solid mesh needs to be interpolated on the fluid mesh in order to update the CFD boundary conditions

$$\mathbf{u}^{(F)} = \mathbf{J}^{(FS)}\mathbf{u}^{(S)} \tag{5.74}$$

as well as the aerodynamic loads produced by the CFD solver on the fluid mesh need to be interpolated on the solid mesh in order to act as a forcing term in the structural mechanics solver

$$\mathbf{f}^{(S)} = \mathbf{J}^{(SF)}\mathbf{f}^{(F)} \tag{5.75}$$

In this work, we are concerned with the discretization of the aerodynamic surface $\Gamma = \partial\Omega\backslash\{\mathscr{S}_0 \cup \mathscr{S}_L\}$. In the following, we compare the properties of two main families of load and displacement transfer algorithms: Consistent interface schemes (section 5.8.1) and conservative interface schemes (section 5.8.2). Next, we introduce the preliminary investigation on the applicability of a meshless conservative interface scheme [136] (section 5.8.3) to the problem at hand. Load consistency issues through fluid–structure interface, not previously reported in other works, have been found during our analysis. For this reason, for the moment we will not proceed with the complete coupled simulation of the fluid–structure problem.

5.8.1 Consistent interface schemes

A consistent interface scheme is required to exactly reproduce a constant displacement *and* pressure over the interface, like in patch tests for contact mechanics [80, 106]. At a discrete level, it can be seen from equations 5.74,5.75 that this requirement is translated in the necessity for both interpolation matrices $\mathbf{J}^{(FS)}, \mathbf{J}^{(FS)}$ to have row-sums equal to one. A number of interpolation schemes can be chosen to this purpose [79], with so-called *area coordinates* (the linear $\mathbb{P}_1$ finite element basis functions evaluation on the surface mesh) being a popular one for topologically conforming surfaces, relying on the localization of surface nodes of one mesh into elements of the other one, and evaluation of linear basis functions on the interface (often easily available in the original codes). As a drawback, typically is not

[2] Vectors $\mathbf{u}, \mathbf{f}$ here denotes the array of nodal values of the continuous fields $\mathbf{u}(\mathbf{x}), \mathbf{f}(\mathbf{x})$.

possible to design simple consistent interface schemes which also conserve the work of the forces acting on the interface [44].

5.8.2 Conservative interface schemes

Conservative interface schemes aim at producing interpolation procedures that allow the conservation of energy between different discretizations of the same underlying topological surface [58, 36, 136]. The idea is to select only an interpolation scheme for the displacement $\mathbf{J}^{(FS)}$, and to let the interpolation of the forces $\mathbf{J}^{(SF)}$ be derived by the equality of the virtual work on the aerodynamic surface computed on the two different fluid (F) and solid (S) meshes

$$\delta\mathscr{L}_{e,\Gamma} = \int_{\Gamma^{(F)}} \delta\mathbf{u}^{(F)}(\mathbf{x}) \cdot \mathbf{f}^{(F)}(\mathbf{x})\, \mathrm{d}\Gamma = \int_{\Gamma^{(S)}} \delta\mathbf{u}^{(S)}(\mathbf{x}) \cdot \mathbf{f}^{(S)}(\mathbf{x})\, \mathrm{d}\Gamma \tag{5.76}$$

Taking into consideration the displacement discretization in equation 5.73, and assuming that we are employing the same discretizations for the surface forces $\mathbf{f}(\mathbf{x})$, computation of mass matrices on the surface Γ lead to

$$\delta\mathbf{u}^{(F)^T}\mathbf{M}_\Gamma^{(F)}\mathbf{f}^{(F)} = \delta\mathbf{u}^{(S)^T}\mathbf{M}_\Gamma^{(S)}\mathbf{f}^{(S)} \tag{5.77}$$

Once a suitable consistent interpolation of displacements $\mathbf{J}^{(FS)}$ from the solid mesh to the fluid mesh is chosen[3], substitution into eq. 5.76 leads to

$$\delta\mathbf{u}^{(S)^T}\mathbf{J}^{(FS)^T}\mathbf{M}_\Gamma^{(F)}\mathbf{f}^{(F)} = \delta\mathbf{u}^{(S)^T}\mathbf{M}_\Gamma^{(S)}\mathbf{f}^{(S)} \tag{5.78}$$

from which it is possible to identify the forces interpolation matrix as

$$\mathbf{J}^{(SF)} \triangleq \mathbf{M}_\Gamma^{(S)^{-1}}\mathbf{J}^{(FS)^T}\mathbf{M}_\Gamma^{(F)} \tag{5.79}$$

For convenience, the vectors of integrated nodal forces are often defined as

$$\begin{aligned} \mathbf{F}^S &= \mathbf{M}_\Gamma^{(S)}\mathbf{f}^{(S)} \\ \mathbf{F}^F &= \mathbf{M}_\Gamma^{(F)}\mathbf{f}^{(F)} \end{aligned} \tag{5.80}$$

so that, using pedices w, f to denote specific nodes on the solid wet surface $\Gamma^{(S)}$ or on the fluid boundary surface $\Gamma^{(F)}$, respectively, the virtual external work is compactly rewritten as a summation of nodal values

$$\delta\mathscr{L}_{e,\Gamma} = \sum_{f=1}^{N_f} \delta\mathbf{u}_f^{(F)}\mathbf{F}_f^{(F)} = \sum_{w=1}^{N_w} \delta\mathbf{u}_w^{(S)}\mathbf{F}_w^{(S)} \tag{5.81}$$

In a practical implementation, a conservative interface scheme is built in the following steps.

[3] Matrix $\mathbf{J}^{(FS)}$, acting on vectors containing nodal values in three space directions, actually replicates three times the same interpolation matrix $\mathbf{I}^{(FS)}$ acting in each space direction

1. **Interpolation** of displacements and load

$$\mathbf{u}_f^{(F)} = \sum_{w=1}^{N_w} I_{fw}^{(FS)} \mathbf{u}_w^{(S)}, \qquad f = 1, \ldots, N_f \tag{5.82}$$

and substitution of this expression into the external virtual work (equation 5.81), so that the integrated nodal forces $\mathbf{F}_w^{(S)}$ can be defined

$$\delta \mathscr{L}_{e,\Gamma} = \sum_{f=1}^{N_f} \delta \mathbf{u}_f^{(F)} \mathbf{F}_f^{(F)} = \sum_{w=1}^{N_w} \mathbf{u}_w^{(S)} \underbrace{\sum_{f}^{N_f} I_{fw} \mathbf{F}_f^{(F)}}_{\triangleq \mathbf{F}_w^{(S)}} = \sum_{w=1}^{N_w} \delta \mathbf{u}_w^{(S)} \mathbf{F}_w^{(S)} \tag{5.83}$$

2. **Assembly** of the displacement and force arrays requires the correct mapping of boundary displacement and forces $u_{\{d,w\}}^{(S)}, f_{\{d,w\}}^{(S)}$, accessed by indices d (space direction index) and w (wet surface node index), in the domain displacement and force vectors $u_{\{i,\{d,s\}\}}^{\Omega}$, accessed by indices d (space direction index) and s (cross-section node index). This is possible once the set of nodes on closed loop constituting the external boundary of each structural cross-section is identified and indexed by the pair of indices i (x-axis index) and l (loop index), then mappings $w = w(i,l)$ and $s = s(l)$ returning the wet-surface index w and the cross-section index s given the x-axis index i and the loop index l are computed.

   This leads to the assembly relations

$$\begin{cases} u_{\{d,w(i,l)\}}^{(S)} &= u_{\{i,\{d,s(l)\}\}}^{\Omega} \\ f_{\{d,w(i,l)\}}^{(S)} &= f_{\{i,\{d,s(l)\}\}}^{\Omega} \end{cases} \tag{5.84}$$
$$d = 1, \ldots, 3, \quad l = 1, \ldots, N_l, \quad i = 1, \ldots, N_x + 1$$

3. **Projection** of the load onto the reduced basis is an additional step required by the reduced-order model. In this case, we aim at assembling the projected load $f_{i,r}^{\Omega,\pi}$ accessed by indices i (x-axis coordinate index) and r (mode index). Taking the reconstruction of the domain displacement (equation 5.50)

$$u_{\{i,\{d,s\}\}}^{\Omega} = \sum_{r=1}^{N_r} P_{\{d,s\}r} q_{\{i,r\}} \tag{5.85}$$

and inserting it in the expression of the virtual work

$$\delta \mathscr{L}_{e,\Gamma}^{\pi} = \sum_{i=1,r=1}^{N_x+1,N_r} \delta q_{\{i,r\}} \underbrace{\sum_{d=1,s=1}^{3,N_s} f_{\{i,\{d,s\}\}}^{\Omega} P_{\{d,s\}r}}_{\triangleq f_{i,r}^{\Omega,\pi}} \tag{5.86}$$

an expression for the projected load is obtained.

The above steps require the definition of a solution interpolation algorithm, represented by matrix $\mathbf{I}^{(FS)}$, but are independent from the specific choice.

Can we have conservative *and* consistent interfaces? In a recent work [44], it has been shown how it is not possible, in general, to preserve consistency of pressure in a conservative interface scheme, even if a consistent displacement interpolation is employed. In fact, no consistency requirement is explicitly enforced on the conservative interpolation matrix $\mathbf{J}^{(SF)}$ (equation 5.79), so consistency of interpolated pressure should be verified for the interface scheme at hand, and it could depend from the choice of the displacement interpolation method. For example, for mortar methods [58] it has been shown [44] that pressure consistency holds only for topologically conforming interfaces. Some theoretical work to overcome the problem in contact mechanics can be found in [170], while a recent conservative *and* consistent coupling has been proposed in [3] through the extension to non-conforming mesh interfaces of the supermesh approach already proposed for conservative mesh adaptation [60, 118].

In the following, we will introduce the conservative interface scheme selected for our analysis (section 5.8.3) and the issues found with the lack of guaranteed consistency (section 5.8.5).

5.8.3 Conservative meshless reconstruction of surfaces.

In this work, we investigate the applicability of the conservative meshless interpolation method proposed in [136] for fluid–structure interaction problems. Meshless methods are a class of numerical methods in computational mechanics aiming at discretizing continuous problem given only its values in distinct points, without an underlying mesh or grid structure [20]. Their most extensive application is in smooth particle hydrodynamics (SPH), solid mechanics involving interfaces (such as fracture propagation, contacts) and partitioned problems [149]

Moving least-square approximation [20] consists in finding an approximant u^h for a scalar function $u(\mathbf{x})$ by locally minimizing the weighted least square error by means of local support kernel functions. Following [103], a local approximation near point $\mathbf{x}$ is defined as

$$u^h(\mathbf{x},\boldsymbol{\xi}) = \sum_{i=1}^{m} p_i(\boldsymbol{\xi})\, a_i(\mathbf{x}) = \mathbf{p}^T(\boldsymbol{\xi})\mathbf{a}(\mathbf{x}) \tag{5.87}$$

where $\mathbf{x}$ is an evaluation point, and $\boldsymbol{\xi}$ are surface coordinates. Vector $\mathbf{p}(\boldsymbol{\xi})$ collects a polynomial basis on $\mathbb{R}^2$, and vector $\mathbf{a}(\mathbf{x})$ represents the unknown in the local

approximation. For each evaluation point, the least square error functional reads

$$J(\mathbf{x}) = \sum_l w(\mathbf{x}-\boldsymbol{\xi}_l)\left(u^h(\mathbf{x},\boldsymbol{\xi}_l) - u(\boldsymbol{\xi}_l)\right)^2 = \sum_l w(\mathbf{x}-\boldsymbol{\xi}_l)\left(\sum_{i=1}^{m} p_i(\boldsymbol{\xi}_l)a_i(\mathbf{x}) - u(\boldsymbol{\xi}_l)\right)^2 \tag{5.88}$$

where $\boldsymbol{\xi}_l$ is a set of surface points. With the definition of the following vector and matrices

$$\mathbf{u}^T \triangleq [u(\boldsymbol{\xi}_1)\dots u(\boldsymbol{\xi}_S)] \tag{5.89}$$

$$\mathbf{P} \triangleq \begin{bmatrix} p_1(\boldsymbol{\xi}_1) & p_2(\boldsymbol{\xi}_1) & \dots & p_m(\boldsymbol{\xi}_1) \\ p_1(\boldsymbol{\xi}_2) & p_2(\boldsymbol{\xi}_2) & \dots & p_m(\boldsymbol{\xi}_2) \\ \vdots & \vdots & \ddots & \vdots \\ p_1(\boldsymbol{\xi}_n) & p_2(\boldsymbol{\xi}_n) & \dots & p_m(\boldsymbol{\xi}_n) \end{bmatrix} \tag{5.90}$$

$$\mathbf{W}(\mathbf{x}) \triangleq \begin{bmatrix} w(\mathbf{x}-\boldsymbol{\xi}_1) & 0 & \dots & 0 \\ 0 & w(\mathbf{x}-\boldsymbol{\xi}_2) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & w(\mathbf{x}-\boldsymbol{\xi}_n) \end{bmatrix} \tag{5.91}$$

the functional is written in matrix form

$$J(\mathbf{x}) = (\mathbf{P}\mathbf{a}(\mathbf{x}) - \mathbf{u})^T \mathbf{W}(\mathbf{x})\,(\mathbf{P}\mathbf{a}(\mathbf{x}) - \mathbf{u}) \tag{5.92}$$

and can be minimized as

$$\frac{\partial J}{\partial \mathbf{a}} = \mathbf{A}(\mathbf{x})\mathbf{a}(\mathbf{x}) - \mathbf{B}(\mathbf{x})\mathbf{u} = \mathbf{0} \tag{5.93}$$

where the following matrices have been introduced for convenience

$$\mathbf{A} \triangleq \mathbf{P}^T\mathbf{W}(\mathbf{x})\mathbf{P}, \qquad \mathbf{B}(\mathbf{x}) \triangleq \mathbf{P}^T\mathbf{W}(\mathbf{x}) \tag{5.94}$$

Finally, the approximation at point $\mathbf{x}$ is recovered as

$$u^h(\mathbf{x}) = \boldsymbol{\phi}^h(\mathbf{x})\mathbf{u} \tag{5.95}$$

where

$$\boldsymbol{\phi}^h(\mathbf{x}) \triangleq \mathbf{p}(\mathbf{x})^T\mathbf{A}(\mathbf{x})^{-1}\mathbf{B}(\mathbf{x}) \tag{5.96}$$

By construction, the approximation is consistent of order k if the basis $\mathbf{p}$ is complete in the polynomials of degree k [20].

5.8.4 Implementation details

Localization algorithm. Due to the peculiarities of the solid geometry under analysis, namely a constant cross-section prism, a specific localization for aerodynamic nodes on the solid wet surface has been devised. Availability of a constant cross-section discretization allows to separate out-of-plane localization from in-plane localization (similarly to the separation of variables employed in the mathematical model), and to skip an explicit construction of a surface mesh on the solid boundary (not available in the data structures). The procedure consists in two steps.

1. **Localization on extrusion axis** is performed by simple one-dimensional binning.
2. **Localization on section boundary** is performed by
   a) Identifying the cross-section boundary elements (one-dimensional segments), by finding the mesh edges not shared by two domain elements.
   b) Grouping boundary elements in *loops* following their adjacency until the same element is accessed twice[4], and identifying the external loop as the one maximizing the enclosed area.
   c) Locating aerodynamic nodes on an external loop element, by parameterizing the loop nodes (and the aerodynamic query node) by the angle between the line connecting it to the cross-section center of mass and an in-plane coordinate axis, so that each aerodynamic node query is performed among the loop element in the (a sort of hashing)[5].

Once this two-steps localization is performed, it is possible to extend the support stencil in both the axial and the cross-section boundary curvilinear direction by selecting adjacent cross sections and loop elements, respectively.

5.8.5 Preliminary results of the application of a meshless conservative interface scheme

In this section, we check the consistency of the force field $\mathbf{f}^{(S)}$ interpolated through the conservative meshless interface scheme introduced in section 5.8.3 by assign-

[4] The cross-section mesh is externally bounded by a closed loop of segments, and can only contain closed loop of segments surrounding internal holes.

[5] When localization on boundary elements is performed by means of area coordinates [65], care should be taken if the query node is in a shadow region (i.e. an angular region not intersecting the prisms generated by two adjacent elements and their outward normal vectors), since this leads the localization procedure to stall.

ing a uniformly constant value on the fluid surface[6]

$$\tilde{\mathbf{f}}^{(F)}(\mathbf{x}) = \mathbf{1}_3 \tag{5.97}$$

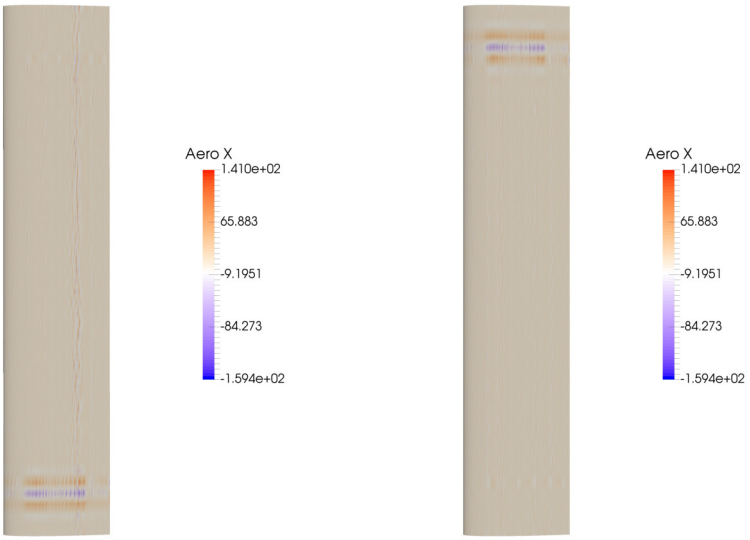
and evaluating the interpolated values $\mathbf{f}^{(S)}$ according the the interpolation matrix defined in equation 5.79, leading to the expression

$$\tilde{\mathbf{f}}^{(S)} = \mathbf{M}_\Gamma^{(S)^{-1}} \mathbf{J}^{(FS)^T} \mathbf{M}_\Gamma^{(F)} \mathbf{1}_{3N_f} \tag{5.98}$$

In order for numerical consistency to be satisfied, the last expression should be constantly equal to 1 in every vector component (thus for every space direction, in every point on the structural interface).

What we find is that the interpolated forces $\mathbf{f}^{(S)}$, given a uniform constant force $\mathbf{f}^{(F)} = \mathbf{1}_{3N_f}$, are indeed equal in the three space dimensions, but the uniform value is not respected, and thus consistency is not verified. Results for the interpolated forces $\tilde{\mathbf{f}}^{(S)}$ are shown in figure 5.20 for one space dimension. The interpolated forces exhibit a strong spurious oscillations of the order of 100 times the original force value, with a wavelength of the order of the grid spacing. and casting doubts on a direct application of this conservative interface scheme to the camber morphing problem, where local load variations are important in determining the projection of the load on eigenvector shapes. Further analysis is required to address this problem, to verify the behaviour of other conservative schemes, or alternatively to investigate the conservation error brought by the usage of a consistent interface scheme.

[6] The expression $\mathbf{1}_m$ is used to denote a constant vector with value 1 in each space direction in $\mathbb{R})^m$.



a Mode 6, 1000×. b Mode 8, 1000×.

Figure 5.20 – Consistency analysis for interpolated surface load $\mathbf{f}^{(S)}(\mathbf{x})$ in the x direction (results are the same in every space direction).

Part III

Numerical simulation of nonclassical aileron buzz

6 Nonclassical aileron buzz simulation over topology-changing dynamic adaptive meshes

In this chapter, the ALE methods over conservative adaptive meshes described in the previous chapter are applied to a case of aeronautical interest, namely the analysis of nonclassical aileron buzz. This application highlights two capabilities of the present method:

1. *Conservative solution transfer among adapted meshes*, necessary to perform an aeroelastic simulation with large relative body motions without introducing a spurious energy dissipation through solution interpolation.

2. *Handling of morphing boundaries*, necessary for the analysis of a two-dimensional continuous wing-aileron configuration.

The present investigation follows several works performed in the Department of Aerospace Science and Technology of Politecnico di Milano [33, 179], from which the geometrical setup and the simulation conditions are choosen.

6.1 Introduction

Unsteady transonic flow. Transonic flow is characterized by the coexistence of subsonic and supersonic flow regions near the surface of the aircraft. Flow deceleration from supersonic to subsonic speed occurs through shock waves, which are nearly normal to the body surface. When the body moves, supersonic flow regions change in size with a consequent movement of shock waves, which can strengthen or disappear partially or completely[158] (fig. 6.1). The unsteady loads brought by this unsteady shock motion are related to some aeroelastic instability phenomena like transonic flutter and buzz [23], which can't be modeled by linear theories due to both the importance of actual geometry and the presence of mixed subsonic–supersonic flow regions [23].

Control-surface buzz. Control-surface buzz is a class of phenomenona characterized by self-sustained oscillations of a control surface on an aircraft wing, typically an aileiron flapping behind an aircraft wing in a transonic flow regime [102].

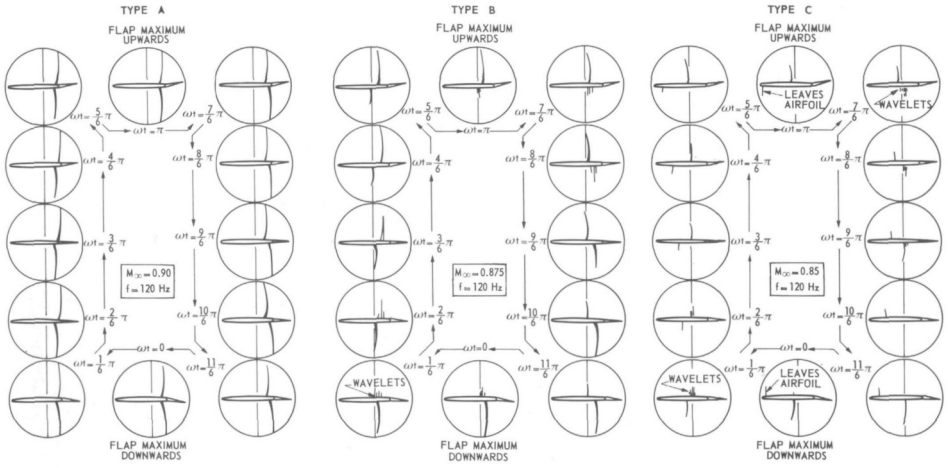


Figure 6.1 – Proposed categories for periodical shock wave motions (from [158]).

As explained in a 1962 report by Lambourne [102], this phenomenon was first observed in high subsonic flight in 1945 on a P-80 jet aircraft [102, 22] and then investigated by wind tunnel tests [54, 53] which showed that the phenomenon can be successfully reproduced by means of a single degree-of-freedom system (a freely-rotating aileron about its hinge) and that it is associated with the backwards and forwards movement of shock waves on the surface of the wing (and possibly of the aileron) occurring with a phase lag with respect to the aileron oscillation. The extent of the local supersonic regions varies as the Mach number is increased, and allows to distinguish three main categories of buzz (fig. 6.2) which are experienced with increasing flight speed [102].

- *Type A* — At a Mach number lightly above the critical Mach number, local supersonic flow regions are present on the surface of the wing only, and shock waves remain well ahead of the control-surface hinge during their oscillating motion. The control-surface is immersed in the shock-induced separated flow region, which appears to be the main source of the instability.

- *Type B* — At higher Mach number the shock wave oscillates in proximity of the control-surface hinge, and the aileron oscillation increase the excursion of the shock wave movement on and off the control-surface. This shock wave position appears to be the main driver for the instability.

- *Type C (trailing-edge buzz)* — The Mach number is high enough to make the supersonic flow region extend over the entire wing surface, and shock waves form at the trailing edge of the control-surface. In this flow regime, the in-

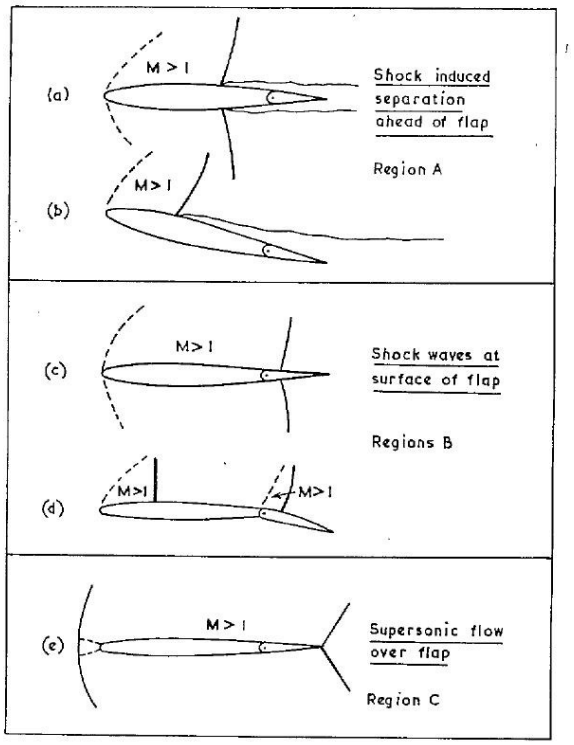
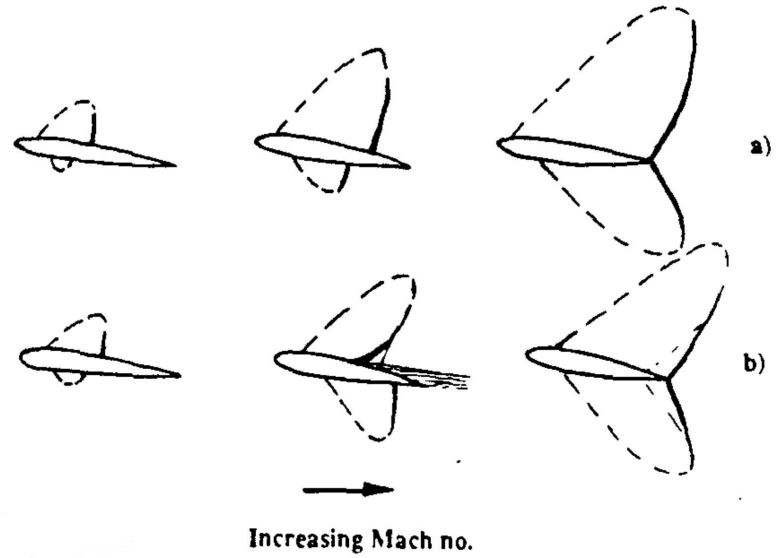


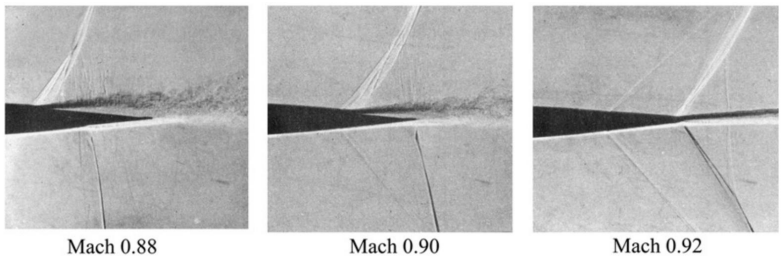
Figure 6.2 – Proposed buzz categories (from [102]).

stability is determined by the negative aerodynamic damping predicted by potential flow theories [102]. This instability disappears as the Mach number is further increased.

Nonclassical aileron buzz. Bendiksen [22] successfully reproduced the last two families of buzz by means of inviscid flow calculations, and labeled them as *nonclassical* in order to stress the fact that they are not determined by shock–boundary layer interaction, differently from *classical* (type A) buzz which instead is determined by shock-induced separation. For nonclassical type B buzz, the onset of the instability appears to be determined by the oscillation in the shock wave location, while shock-induced flow separation influence the phenomenon by slowing down and stopping the aft movement of the shock wave. This effect can be already appreciated by studying steady-state shock location on an airfoil at moderate angles of attack for increasing Mach number (fig. 6.3)[22, 23, 158]. While in an inviscid flow both the upper and the lower shock would continue to move aft with increasing Mach number, in a viscous flow shock-induced separation would impede the aft movement and even reverse it. Due to these effects, Bendiksen [22]



a Sketch of shock location with increasing Mach number: a) Inviscid flow. b) Viscous flow. (From [22])



b Shock reversal with increasing Mach number in viscous flow (from [22] and [130])

Figure 6.3 – *Shock reversal* in viscous flow.

suggests to carry out comparisons between inviscid flow simulations and wind tunnel tests at the same shock location, rather than at the same Mach number.

Regarding the difference between *buzz* and *flutter*, for a control-surface the term *buzz* denotes the onset of self-sustained oscillations developing into a limit cycle, while the term *flutter* is generally used for diverging oscillations eventually leading to aileron failure [30] (for wings, often the term flutter is used to indicate both limit cicle and explosive oscillations [23]). A clear distinction between the phenomena of limit cycle and diverging oscillations relies on the *stability* of a limit cycle solution, which can be influenced by initial conditions, test conditions or simulation methods [23]. Flight tests and wind tunnel experiments can provide different results in the same nominal flight conditions [30]; for safety purposes, it

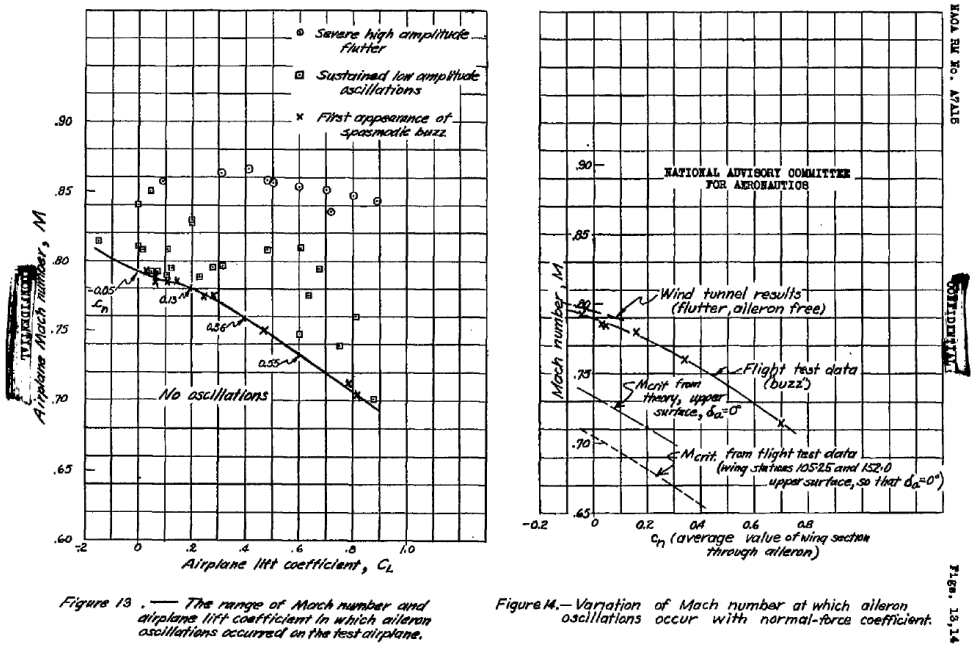


Figure 6.4 – Buzz boundary from flight tests (from [30]).

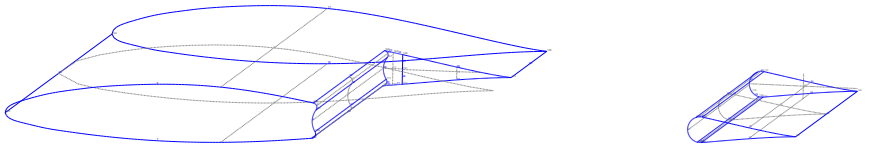
is suggested to always use the buzz boundary individuated by wind tunnel tests (as in fig. 6.4) as an indicator of possible flutter onset at any Mach number greater or equal to the tested one [30].

Test case and objectives. Following [33, 54], we aim at studying the nonclassical type B aileron buzz with a model of a P-80 aircraft wing. Limit cycle oscillations are reported for this configuration at $M = 0.83$. We perform:

- *Three-dimensional simulations* for different aileron spans, in order to provide an assessment of three-dimensional effects.
- *Two-dimensional simulations* with and without wing–aileron gap, in order to provide a first assessment of the effects of geometrical modeling.

6.2 Geometrical setup

We start from the same case studied in [33], with a NACA$65_1213(a = 0.5)$ airfoil wing in a $M = 0.83$ flow. Due to the low tapering of the wing, we simplify the analysis by considering a straight wing enclosed between two wind tunnel walls. The slip boundary conditions in an inviscid compressible flow model is equivalent to imposing a simmetry condition on side walls.



a Wing. b Aileron.

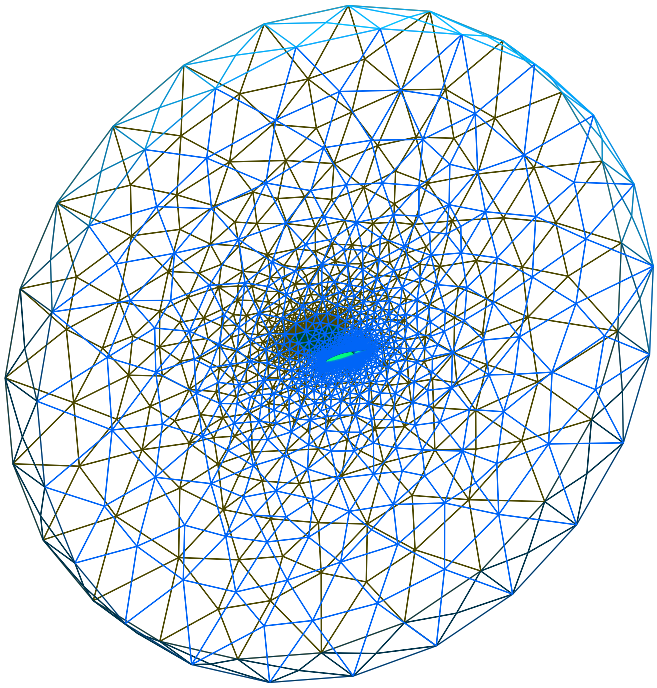
Figure 6.5 – Three-dimensional geometrical model.

Three-dimensional model. Figure 6.5 shows the separate geometrical models for three-dimensional wing and aileron, made with the GMSH software [70]. The wing has a nondimensional chord length $c = 1$, and total span $L = 1.5$. The aileron hinge axis is set at a horizontal distance $x_h = 0.75$ from the wing leading edge. The vertical position of the aileron hinge is chosen so that is at the same vertical distance from the upper and lower surface, thus $y_h = 0.00854$ above the line connecting leading and trailing edge. In this way, the aileron front surface is made from a circular arc with center in the aileron hinge axis. Since its connection with the upper and lower surface is sharp, a fillet operation is used to smoothen both the connections.

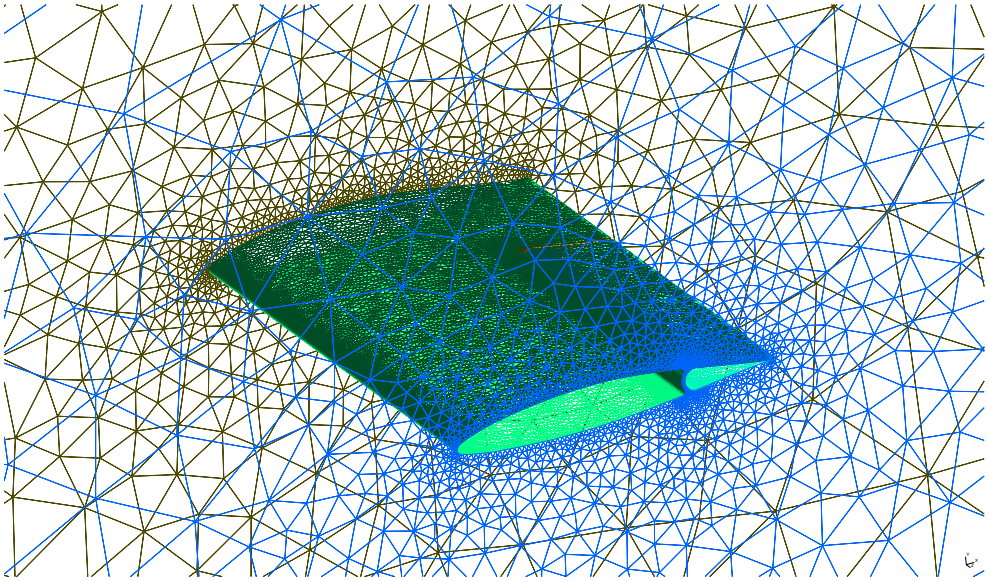
The aileron cut inside the wing is created according to the same procedure, starting with a circular cut centered in the hinge axis, with an additional gap of width $g = 0.01$ between wing and aileron. The side cut allows for an aileron span $b = 1.0$ and is realized with the same gap width $g = 0.01$.

Figure 6.6 shows the initial mesh for the compound wing–aileron configuration.

Two-dimensional model. For two-dimensional analyses over a discontinuous wing–aileron configuration, with an air gap between the two solid bodies, the same cutting procedure used for the 3D model is employed (fig. 6.7a,6.7b). In order to study also a continuous wing–aileron configuration, without air gap between the two solid bodies, two regions on the upper and lower airfoil surface are identified (orange and cyan lines in fig. 6.7c,6.7d), in order to move these strips in time according to the shape interpolation procedure described in section 4.2.1, driven by the aileron angle $\beta(t)$.



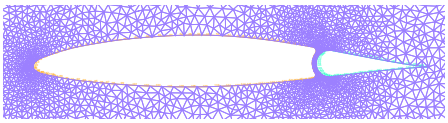
a Three dimensional initial mesh.



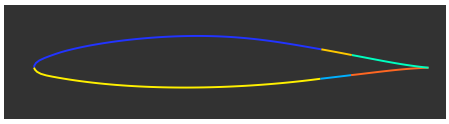
b Zoom on wing and aileron surface mesh.

Figure 6.6 – Three-dimensional initial mesh, with zoom (93729 domain nodes and 528921 elements, minimum edge size $h_{\min} = 0.0015$).

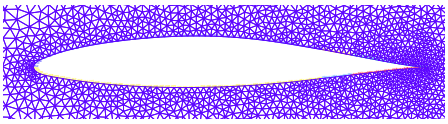
a 2D model with gap.



b 2D initial mesh with gap.



c 2D model without gap.



d 2D initial mesh without gap.

Figure 6.7 – Two-dimensional geometrical model and initial mesh, with and without gap between wing and aileron.

Rigid body model We recall here the nondimensional equation of motion for a rotating rigid aileron introduced in section 4.1.1

$$\hat{J}\frac{\mathrm{d}^2\beta}{\mathrm{d}t^2}+\hat{C}\frac{\mathrm{d}\beta}{\mathrm{d}t}+\hat{K}\beta=C_M^f \tag{6.1}$$

For the free body rotation problem, the equation is solved for $\hat{C}=\hat{K}=0$. Following reference [86] the aileron has inertia per unit span $\tilde{J}=0.24217$ kg m. The nondimensional inertia $\hat{J}$, in three-dimensions, is found as

$$\hat{J}=\frac{\tilde{J}b}{q_\infty c^3\mathscr{T}^2} \tag{6.2}$$

where $q_\infty=31097$ Pa, and the time constant $\mathscr{T}=0.00544$ s is set from the dimensional chord $c=1.472$ m and the flight speed $V_\infty=270.5$m/s.

6.3 Validation

6.3.1 Infinite-span wing-only steady simulations

For validation purposes, we start from a three-dimensional wing-only configuration of nondimensional chord $c=1$ and nondimensional span $L=1.5$, enclosed between two symmetry walls. This configuration is representative of an infinite-span wing, and pressure coefficient results are compared with available experimental data from reference [30] for Mach number $M=0.22$ in fig. 6.8.

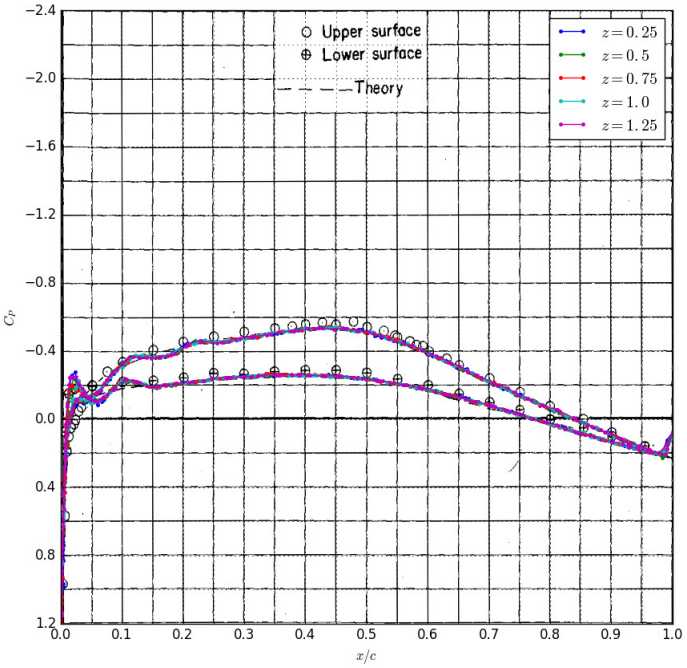


Figure 6.8 – Comparison of computed and experimental pressure coefficients (data and image from [30]) at $M = 0.22$, for different locations z along the wing span.

6.3.2 Three-dimensional imposed aileron oscillation over adaptive meshes

As a validation step, we solve the aileron nondimensional equation of motion

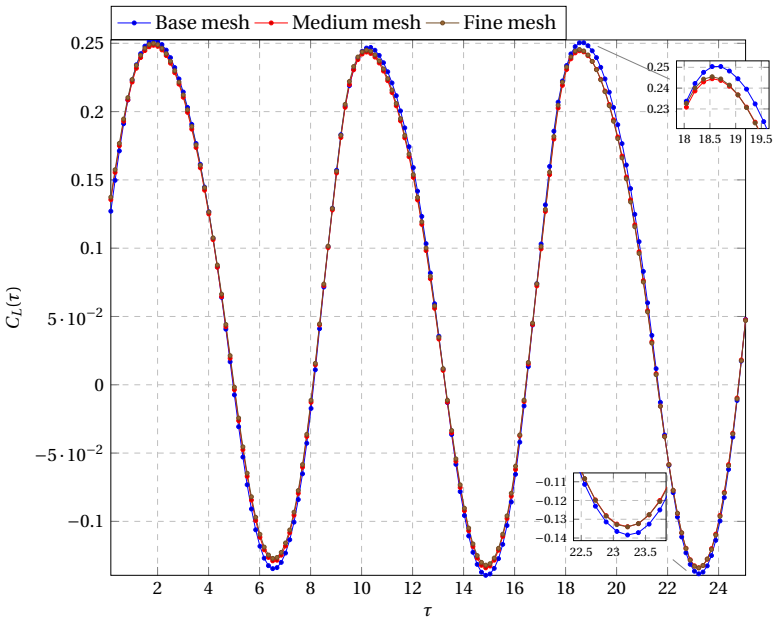
$$\hat{J}\frac{\mathrm{d}^2\beta}{\mathrm{d}\tau^2} + \hat{K}\beta = 0 \tag{6.3}$$

with initial conditions $\beta_0 = 0$ and $\frac{\mathrm{d}\beta}{\mathrm{d}\tau}|0 = \kappa B$ at the beginning of each solver time step. This allows to reproduce an harmonic response with reduced frequency κ and amplitude B

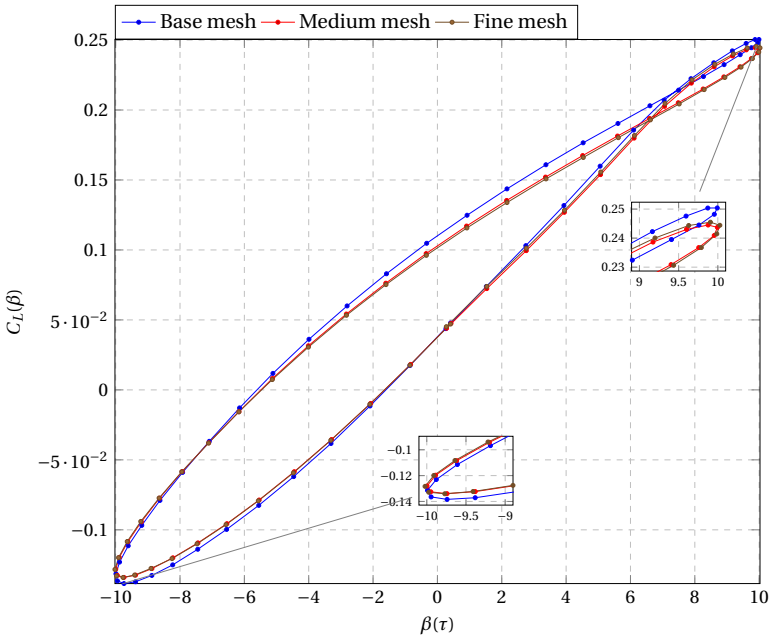
$$\beta(\tau) = B\sin(\kappa\tau) \tag{6.4}$$

The stiffness term is treated as an external forcing, in order to verify also the behavior of the predictor–correction scheme as shown in section 4.1.3.

Comparison for different mesh refinement parameters are performed for lift coefficient (fig. 6.9), moment coefficient(fig. 6.10), mesh nodes and elements number (fig. 6.11). Convergence of the moment coefficient has shown to be trickier than convergence on lift coefficients, as differences show up not only in oscillation peak values, but also in the shape of plot.

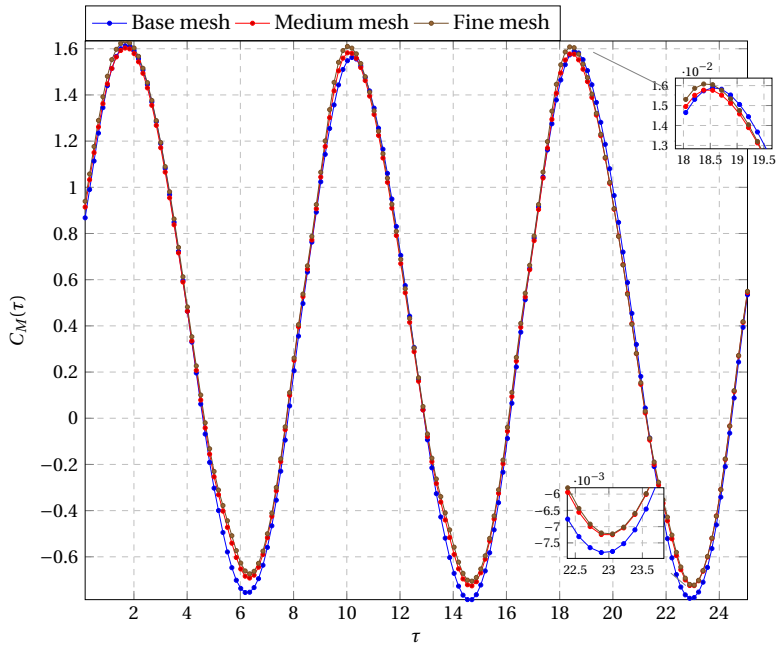


a Time history.

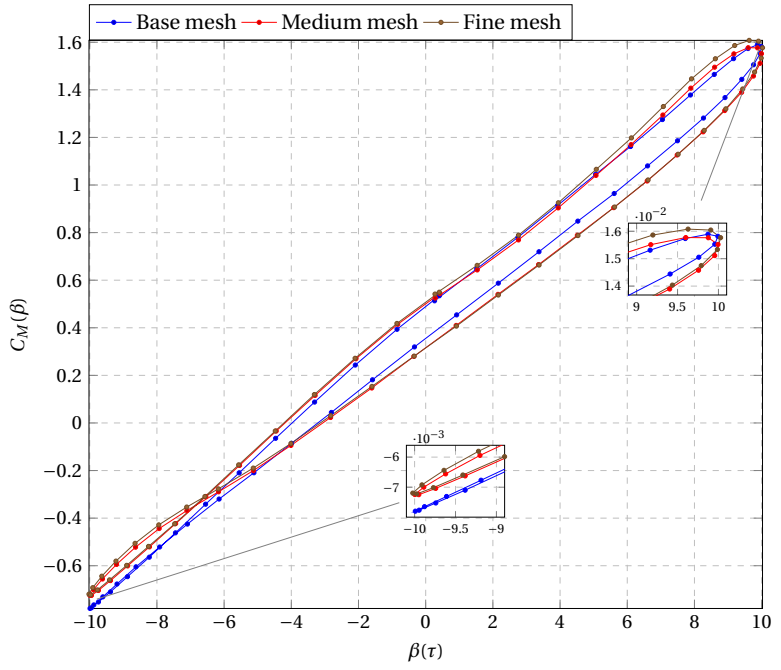


b Phase plot (last oscillation cycle).

Figure 6.9 – Lift coefficient history and phase plot for 3D imposed oscillation case.

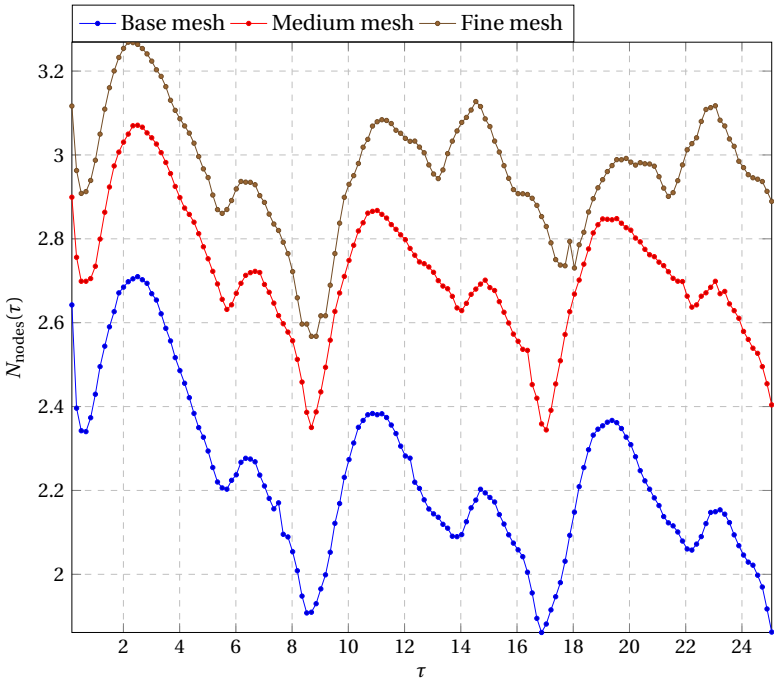


a Time history.

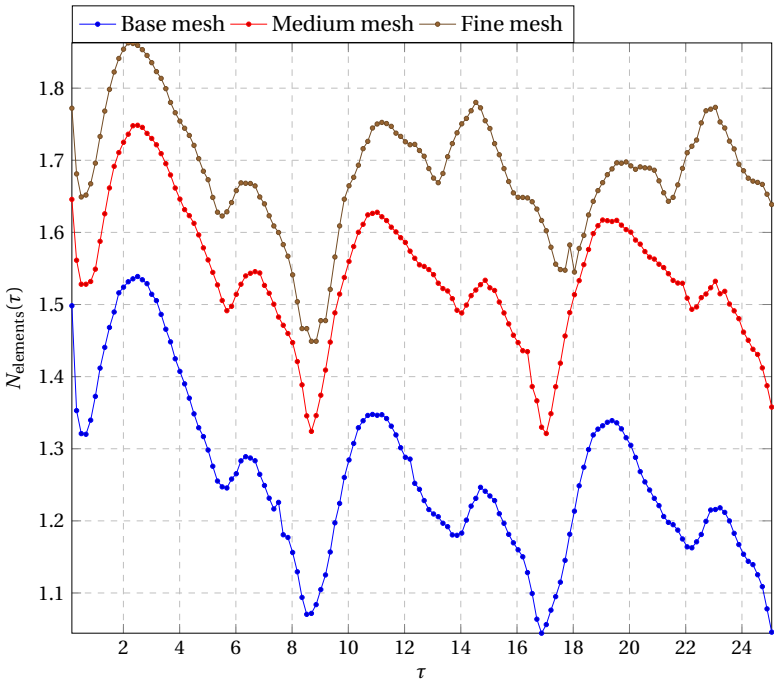


b Phase plot (last oscillation cycle).

Figure 6.10 – Moment coefficient history and phase plot for 3D imposed oscillation case.



a Number of mesh nodes for 3D imposed oscillation case.



b Number of elements for 3D imposed oscillation case.

Figure 6.11 – Nodes and elements number history for 3D imposed oscillation case.

6.4 Two-dimensional buzz simulations over discontinuous and continuous wing–aileron configurations

In this section, we compare the discontinuous and continuous wing–aileron configurations (with and without air gap) shown in fig. 6.7. We consider the free oscillation of the aileron according to the equation

$$\hat{J}\frac{\mathrm{d}^2\beta}{\mathrm{d}\tau^2} = C_M^f(\beta) \tag{6.5}$$

The equation is coupled with the flow solver through the predictor–corrector procedure described in section 4.1.2.

Moment coefficient and aileron angle time histories (fig. 6.12) show a dramatic difference between the two configurations, with oscillations which are completely damped when a continuous surface is present, contrary to the discontinuous configuration.

Analysis of shock waves motion. The difference in the flow field for the two geometrical configurations can be appreciated by analysing the Mach number distribution (fig. 6.13). In the discontinuous configuration, the curvature of the gap corners on the back of the wing imposes an abrupt deceleration of the flow in the downstream direction, making the each corner an upper limit for the aft movement of shock waves. If the flow accelerates again downstream beyond these points, new shocks will form on the surface of the aileron. The flow inside the gap appears to change in direction between upstroke and downstroke aileron motions (fig. 6.14), in phase with the formation of new shocks on the lower or upper surface of the aileron. This mechanism appears to take place in each buzz condition with discontinuous bodies, both in 2D and in 3D.

On the other hand, in the continuous configuration the initial shock location is already different from the one over the discontinuous configuration, suggesting that the buzz boundary could be different [22]. In fact, starting from an initial condition $\beta_0 = 0°$, in the current analysis buzz oscillations are found again at $M = 0.85$ (fig. 6.16). Buzz oscillations can be found for a lower Mach number if the initial condition is changed, as it is the case when $\beta_0 = -6°$ at $M = 0.84$ (fig. 6.17). This behaviour is consistent with the nonlinearity of the mathematical model, for which the stability of solutions depends on the initial conditions [155, 82]. In the present case, the system "switches" from a fixed point solution to a limit cycle oscillation as the initial deflection angle is increased.

In this configuration, there is no flow leakage from upper to lower wing surface (and vice versa) impeding shocks from travelling all over the upper or lower

surface, but changes in surface curvature play a role in modifyng the shock pattern. Since the nonlinear shape interpolation procedure presented in section 4.2.1 is sufficient to get an analytical expression for the surface movement with arbitrary degree of continuity, while preserving only C^0 continuity at the end points (a problem shared by other PDE-based methods, as outlined in section 4.2.1), the end points act as triggers for an expansion fan (during downstoke movements) or a shock (during upstroke movements) upstream of the already present traveling shock. This results, respectively, in the delta shock pattern and the expansion fan–shock interaction showed in fig. 6.15.

6.5 Three-dimensional buzz simulations

In this section, the free oscillation of the three-dimensional aileron is considered. Our aim is to highlight the specific effects which are present in a three-dimensional model, which are lost in a typical two-dimensional simulation. As before, we consider the free oscillation of the aileron according to the equation

$$\hat{J}\frac{\mathrm{d}^2\beta}{\mathrm{d}\tau^2} = C_M^f(\beta) \tag{6.6}$$

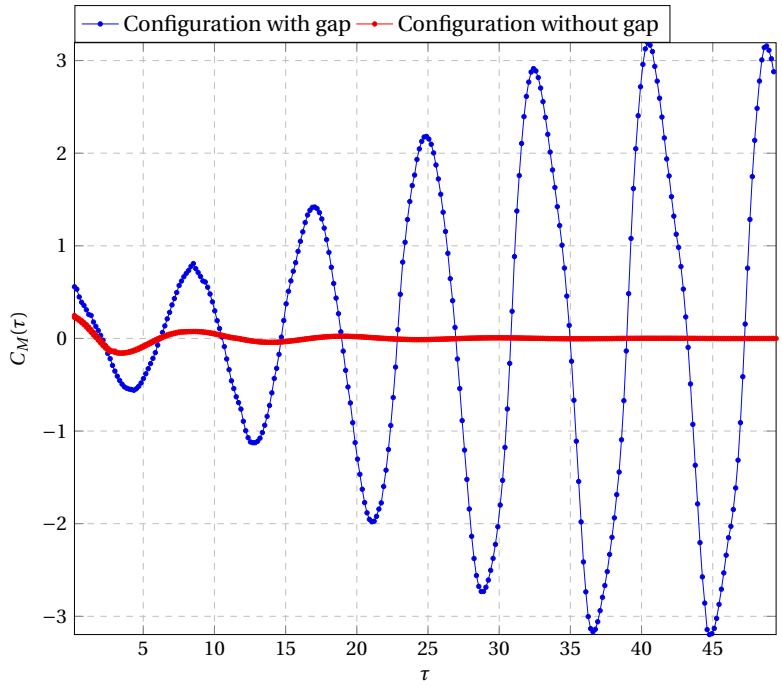
coupled to the flow solver through the predictor–corrector procedure described in section 4.1.2.

6.5.1 Finite and infinite aileron span

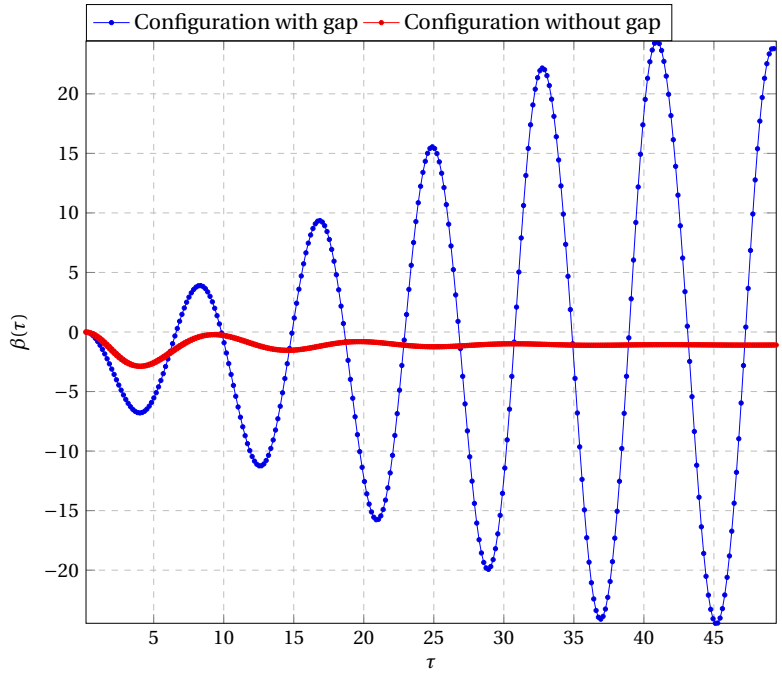
The comparison between a finite-span and an infinite-span (quasi-2D) aileron configuration (fig. 6.18) is shown in fig. 6.20). Results for the quasi-2D configuration are consistent with two-dimensional results, and show a decrease in frequency when a finite-span aileron is considered. This issue is further investigated in the following section.

6.5.2 Varying finite aileron span

Since the comparison between finite-span and infinte-span aileron showed a discrepancy in oscillation frequency, we investigate the frequency trend as the finite aileron span is changed. We test four configurations, shown in fig. 6.21, for progressively shorter ailerons. The span of the wing root is kept fixed. Moment coefficient results (fig. 6.22a) show that the amplitude of the moment coefficient oscillations are progressively smaller, consistently with the reduced size of the aerodynamic surface, while aileron angle oscillations (fig. 6.22b) converge quicker to a limit cycle. Oscillation frequencies decrease as the aileron span is reduced, confirming the observed difference between the 3D and 2D cases. Reduced and

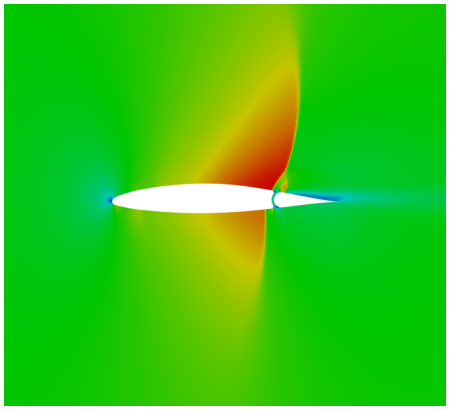


a Moment coefficient time history.

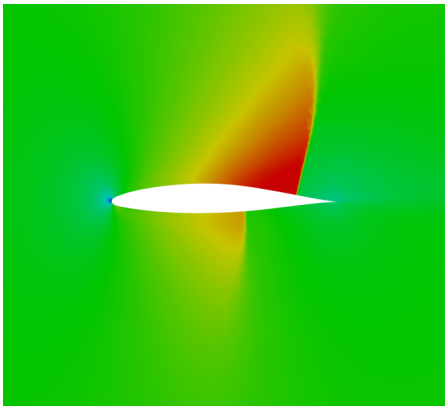


b Aileron angle time history.

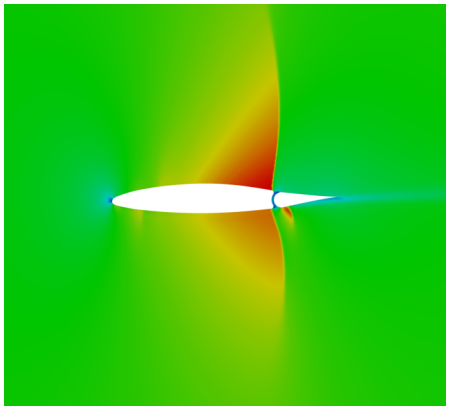
Figure 6.12 – Moment coefficient and aileron angle history with and without wing–aileron gap.



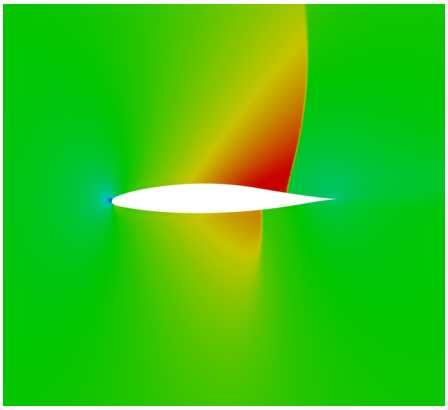
a Discontinuous configuration, $\tau = 0$.



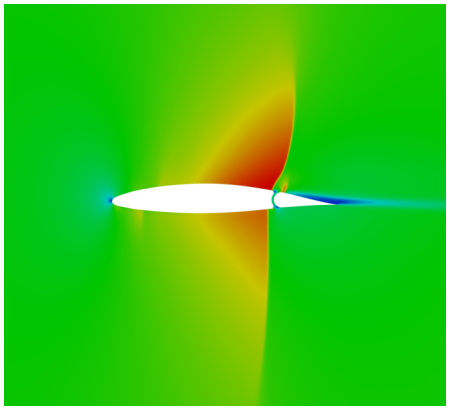
b Continuous configuration, $\tau = 0$.



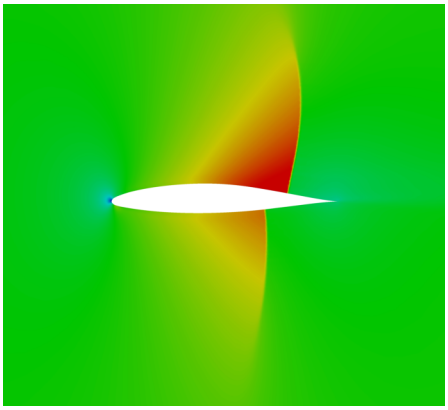
c Discontinuous configuration, $\tau = 4.51062$.



d Continuous configuration, $\tau = 4.51062$.

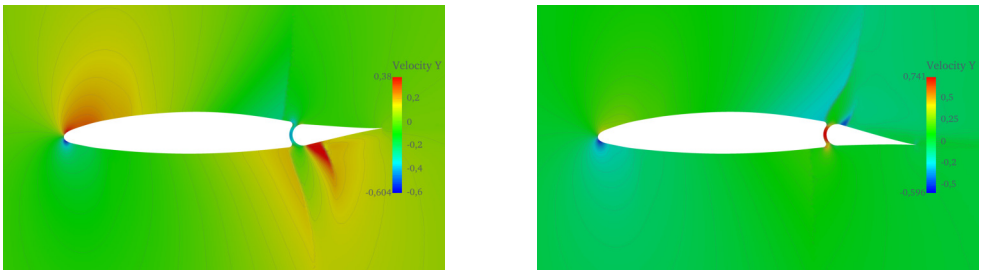


e Discontinuous configuration, $\tau = 8.52006$.



f Continuous configuration, $\tau = 8.52006$.

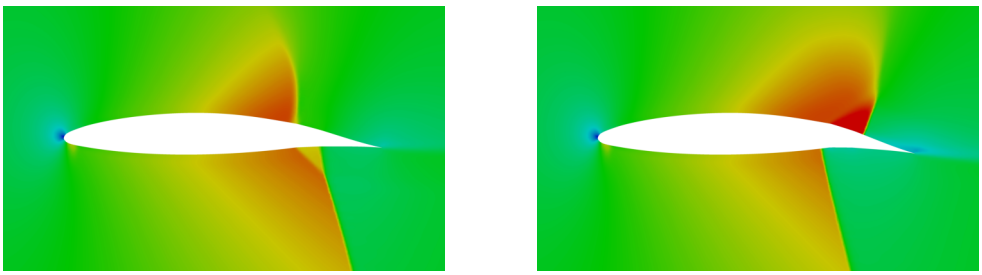
Figure 6.13 – Comparison of Mach number distributions at $M = 0.83$ for the continuous and discontinuous wing–aileron configurations.



a Aileron span $b = 1.0$, wing span $L = 1.5$.

b Aileron span $b = 0.75$, wing span $L = 1.25$.

Figure 6.14 – Vertical velocity at $M = 0.83$ for the discontinuous wing–aileron configuration, in a peak aft movement of the lower and upper shock.



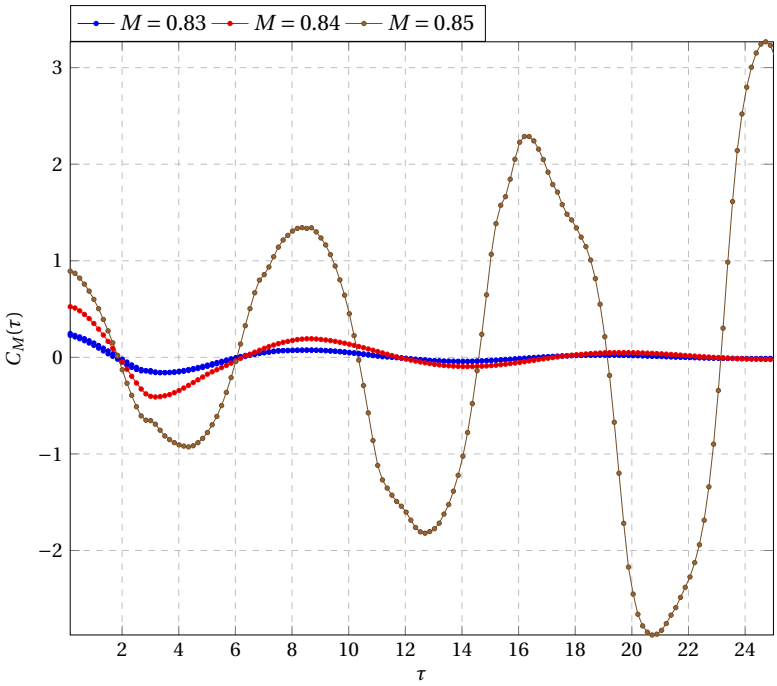
a Aileron span $b = 1.0$, wing span $L = 1.5$.

b Aileron span $b = 0.75$, wing span $L = 1.25$.

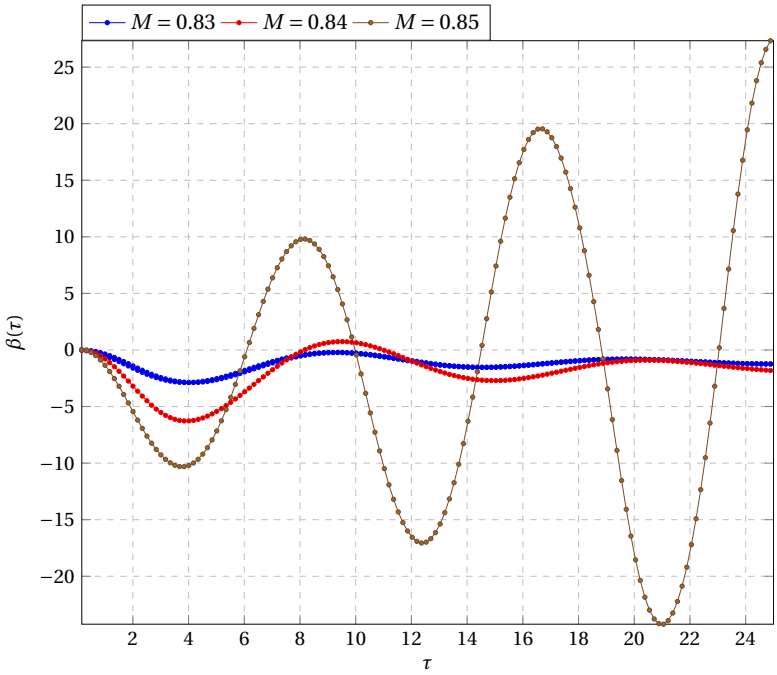
Figure 6.15 – Mach number at $M = 0.84$ for the continuous wing–aileron configuration (starting with $\beta_0 = -6°$), in a peak aft movement of the lower and upper shock.

physical frequencies for different aileron spans are reported in table 6.1. The computed values at $M = 0.83$ match quite well the ones found in experiments. In fact, flight tests [30] reported a frequency of $\simeq 28$ Hz in the Mach range $M = 0.8 - 0.86$, while wind tunnel tests on a full scale wing [54] reported the frequency range $19.4 - 21.2$ Hz at $M = 0.8$. The two-dimensional viscous potential computations in [86] also found a consistent frequency of 21.67 Hz at ($M = 0.8243$). Some comments on the effects of the variation in aileron span on the fluid flow and the shock wave pattern is given in the next paragraph.

Analysis of shock wave motion. Analysis of shock waves motion (fig. 6.23) shows that the effect of the spanwise air gap between wing and aileron is that of imposing an abroupt flow deceleration at the end of the wind surface through a first shock wave, with a new acceleration of the flow on the aileron surface causing a second shock wave to appear during sufficiently wide upstroke or downstroke aileron motion. This re-acceleration is not possible inside the chordwise wing–aileron air gap, so the aileron shock wave always starts from the aileron front corner

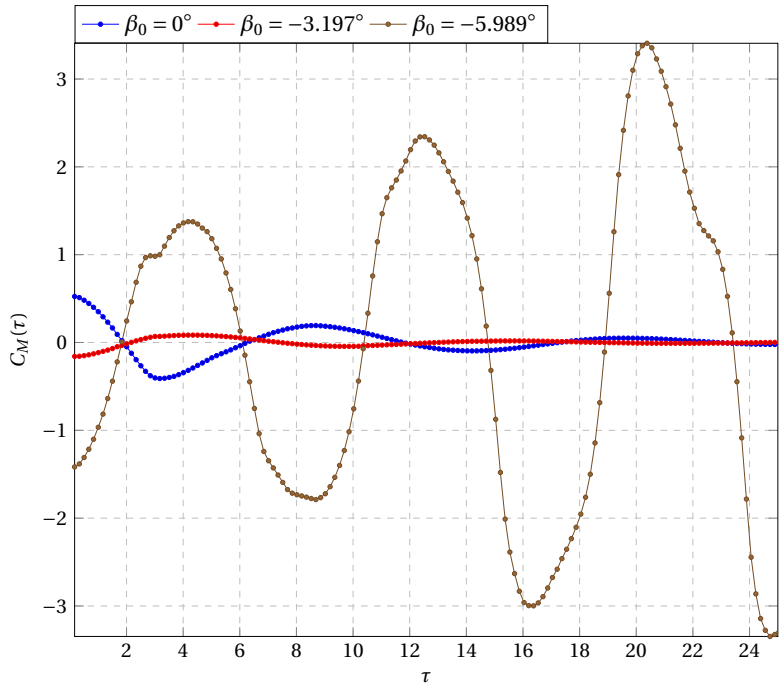


a Moment coefficient time history.

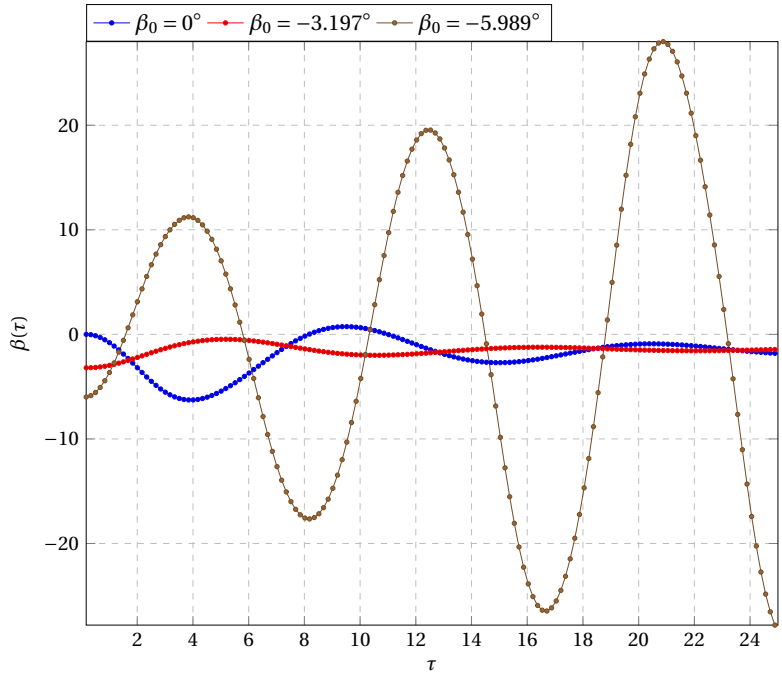


b Aileron angle time history.

Figure 6.16 – Moment coefficient and aileron angle history without wing–aileron gap, varying Mach number, with initial condition $\beta_0 = 0°$.

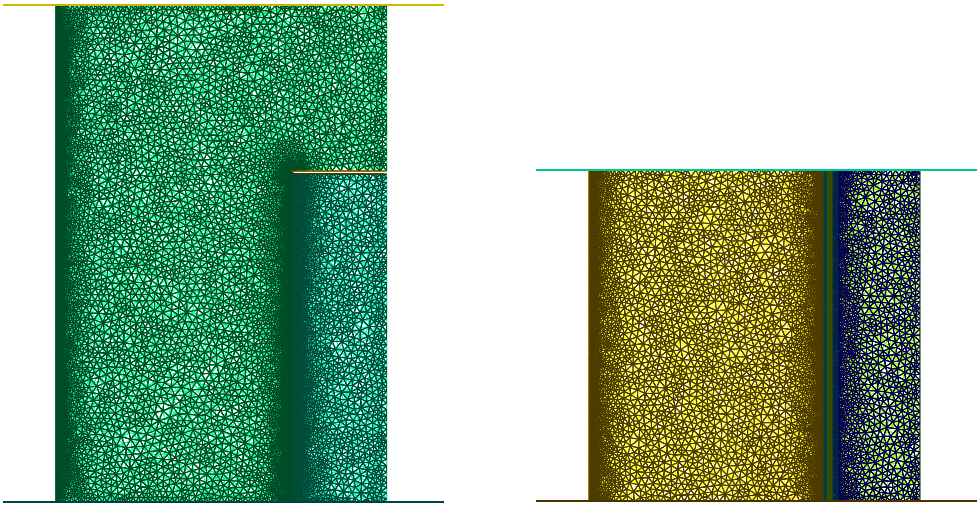


a Moment coefficient time history.



b Aileron angle time history.

Figure 6.17 – Moment coefficient and aileron angle history without wing–aileron gap, varying initial condition, at $M = 0.84$.



a Aileron span $b = 1.0$, wing span $L = 1.5$.

b Aileron span $b = 1.0$, wing span $L = 1.0$.

Figure 6.18 – Three-dimensional initial mesh with finite-span and infinite-span aileron, top view.

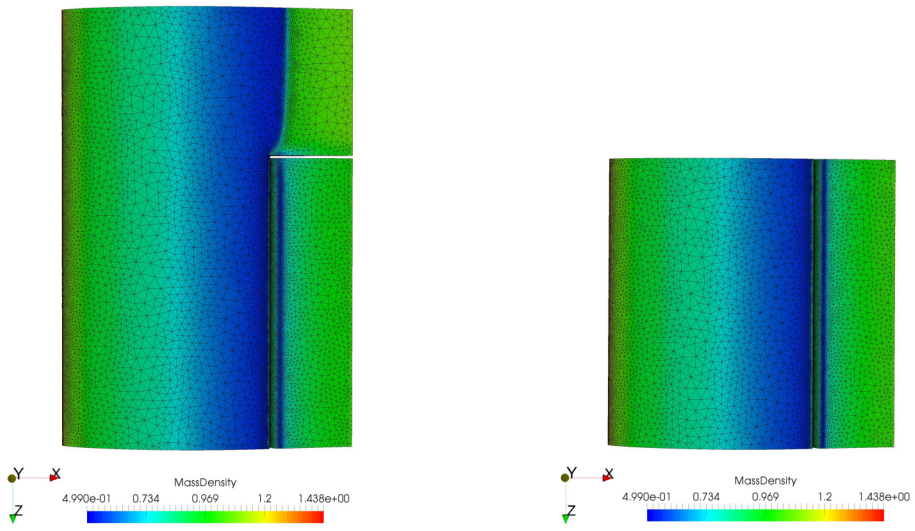
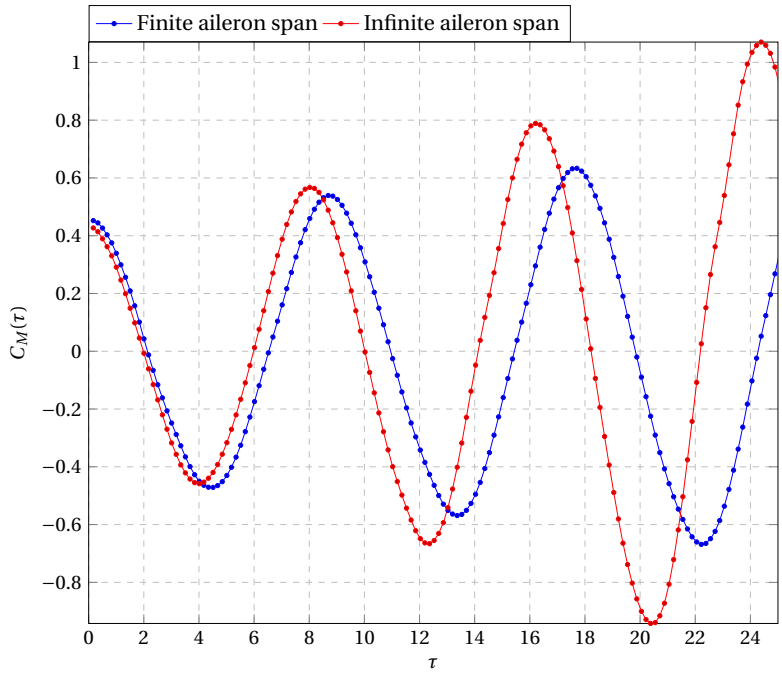
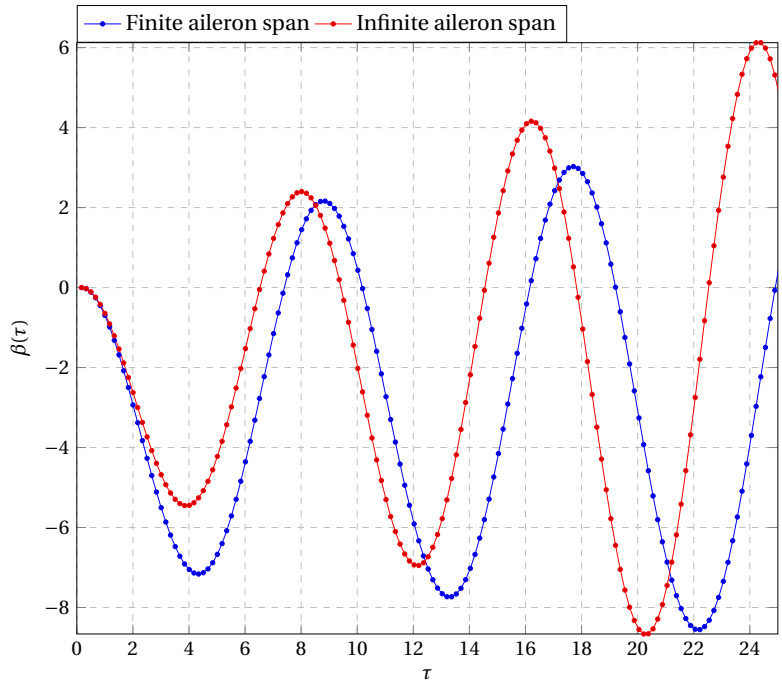


Figure 6.19 – Three-dimensional initial mass density field with finite-span and infinite-span aileron, top view.



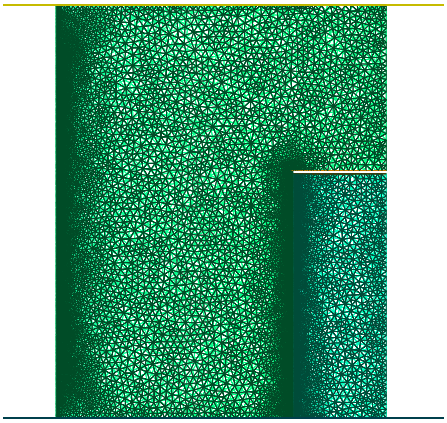
a Moment coefficient time history.



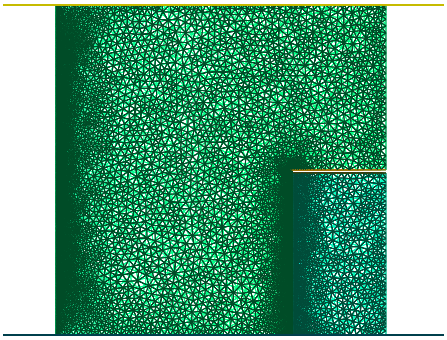
b Aileron angle time history.

Figure 6.20 – Moment coefficient and aileron angle history with finite and infinite aileron span.

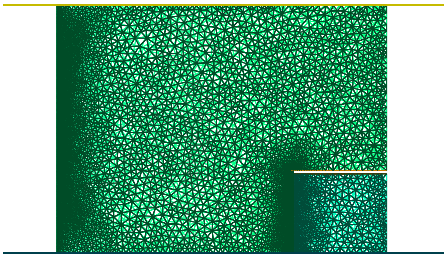
a Aileron span $b = 1.0$, wing span $L = 1.5$.



b Aileron span $b = 0.75$, wing span $L = 1.25$.

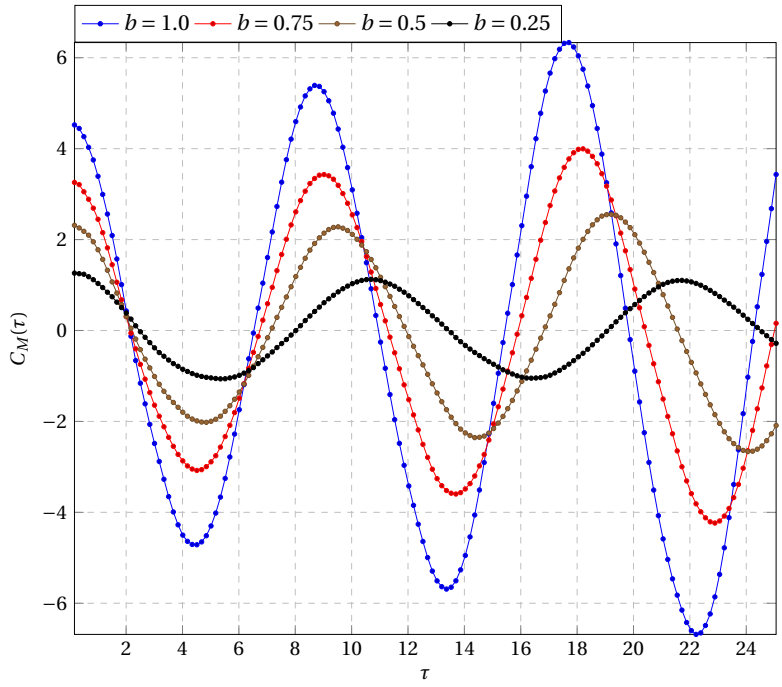


c Aileron span $b = 0.5$, wing span $L = 1$.

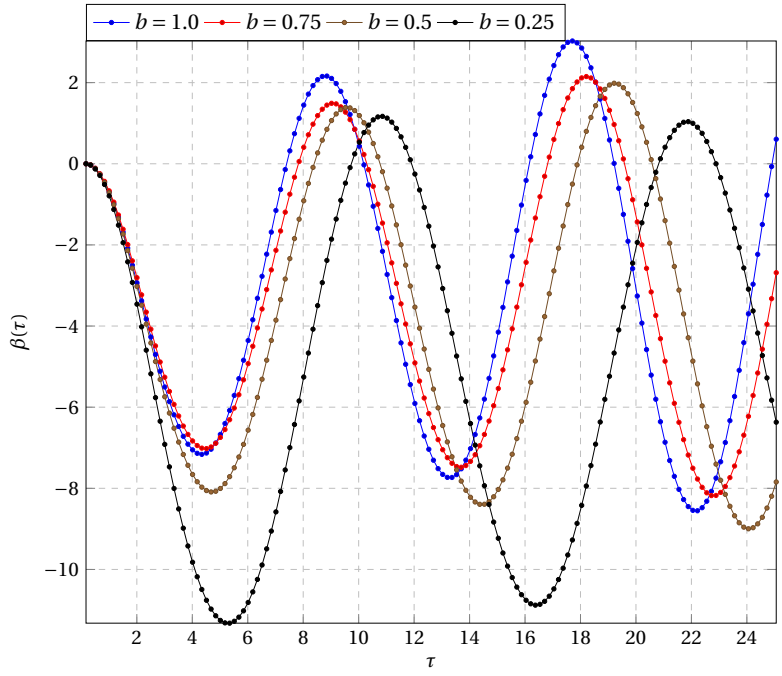


d Aileron span $b = 0.25$, wing span $L = 0.75$.

Figure 6.21 – Three-dimensional initial mesh with varying aileron span (top view).

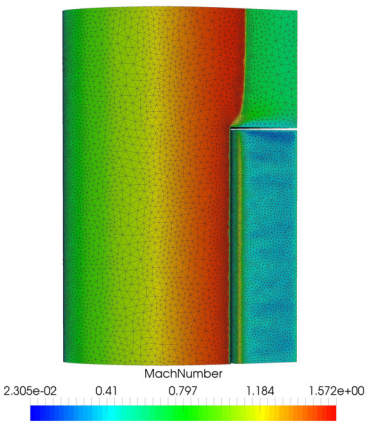


a Moment coefficient time history.

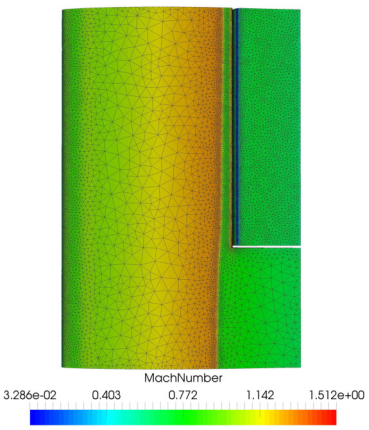


b Aileron angle time history.

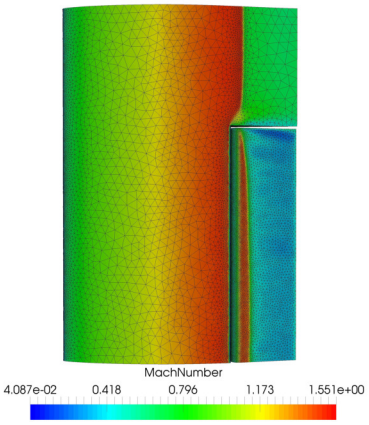
Figure 6.22 – Moment coefficient and aileron angle history with varying aileron span.



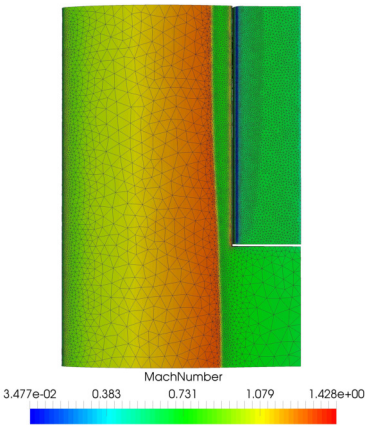
a Initial condition (top view).



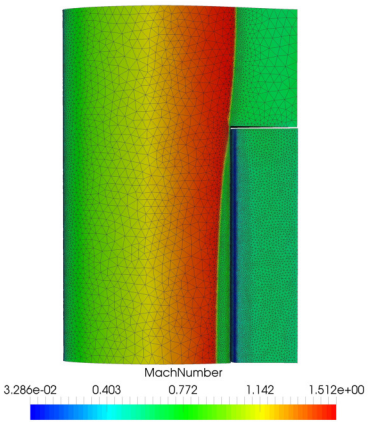
b Initial condition (bottom view).



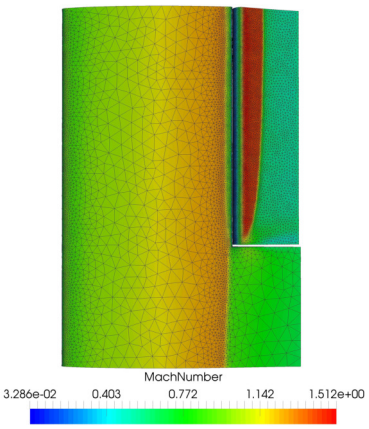
c Downstroke (top view).



d Downstroke (bottom view).



e Upstroke (top view).



f Upstroke (bottom view).

Figure 6.23 – Mach number distribution on wing surface, wing span $L = 1.5$, aileron span $b = 1.0$. Left: Top view. Right: Bottom view.

b/c	$\kappa = \omega c/V$	f
∞	0.78540	22.970
1.0	0.71808	21.002
0.75	0.69813	20.418
0.5	0.66139	19.344
0.25	0.62832	18.376

Table 6.1 – Variations in oscillation frequency with aileron span.

and follows a curved pattern on the aileron surface, where the flow acceleration is stronger. This curved shock pattern influences aileron pressure load as its span is varied, providing an explaination for the variation in the oscillation frequency and amplitude.

7 Conclusions and outlook

7.1 Conclusions

This work presents a conservative solution-transfer methodology for unsteady mesh adaptation in aeroelastic simulations. It constitutes an aeroelastic extension of the conservative methodology for dynamic mesh adaptation orginally introduced in [75, 94, 140] for unsteady compressible fluid flows. The conservative solution-transfer algorithm relies on the computation of the volumes swept by moving cell interfaces during the adaptation phase, both with and without topology change, in order to compute the interface velocities needed in the Arbitrary Lagrangian–Eulerian formulation of the flow equations from a Discrete Geometric Conservation Law. Creation and deletion of interfaces in mesh adaptation with topology change is handled through a continuous interpretation of the node insertion/removal operation, allowing to track in time the volumes swept by cell interfaces as inserted (or removed) cell expands (collapses) from (to) a null volume. An optimization of the `Flowmesh` software implementation has been performed in order to speed up the conservative adaptation procedure, which is performed through the link with the `MMG` remeshing library [42, 45]. Although a strong coupling through callback function is needed to track each local mesh modification in time and to compute swept volumes, the conservative solution transfer procedure is independent from the specific mesh adaptation strategies and thus from the specific software implementation.

A first three-dimensional aeroelastic application is shown with the simulation of the nonclassical aileron buzz [23, 30] on an untapered NACA$65_1$213($a = 0.5$) wing. The three-dimensional simulation on body-fitted meshes is made possible by mesh adaptation, which allows to follow large relative body motions while maintaining a valid mesh. The analysis of different aileron spans highlights that three-dimensional effects related to the air gaps between wing and aileron have indeed an influence on the buzz phenomenon, most importantly on the frequency of the oscillations. The handling of dynamic boundaries through geometry parameterization allows to compare two-dimensional configurations with and without structural continuity between wing and aileron, showing that the presence of

an air gap (and its geometrical modeling) has indeed an influence on the dynamics of the system.

Several building blocks of an unsteady aeroelastic simulation have been considered in this work, leading to complementary results about mesh mechanics, morphing boundaries, structural mechanics and fluid–structure interaction.

- Comparisons of mesh adaptation with and without topology change (r-adaptation) have been performed. In this context, a new model for mesh motion has been implemented in the `FMG` library developed at the INRIA institute in Bordeaux, based on the blending of the well-developed Laplacian model with a linear elastic model. While the Laplacian model is nonlinear and has already been successfully applied to compressible flows [38], the linear elastic model allows a faster iterative solution but has limited applicability for adaptation purposes. A mixed model has been tested on the forward facing step case [176], showing its capability of relieving excessive element stretching (and potentially mesh tangling) brought by the Laplacian model in the normal direction to moving fronts. Preliminary application to an extruded forward facing step shows that these conclusions still hold in three dimensions, where the Laplacian model alone is even more prone to produce excessive element stretching.

- A simple coupling of the conservative mesh adaptation framework with morphing boundaries is proposed by extending the geometric parameterization of the moving boundary to the time domain. A time parameterization of the fixed-topology geometry control points allows to formulate a continuous model for the moving boundary, and to treat variable-topology boundary mesh points as evaluation points for the geometric model. The approach has been successfully tested for a partial and full flap opening on a two-dimensional airfoil, and later applied for the non-classical buzz analysis on a two-dimensional continuous wing-aileron configuration.

- A structural mechanics model has also been developed for the morphing behavior of untapered, untwisted helicopter blades. The generalized beam model proposed by [125] has been extended by the identification of a suitable function basis for a projection-based reduced order model.

- Preliminary analysis of fluid–structure interface schemes has shown that conservative meshless interface schemes do not automatically guarantee consistency of the interpolated pressure, which in the present case is important to preserve local smoothness in the aerodynamic load so to have a reliable evaluation of modal projections.

7.2 Outlook

The above discussion shows the capability of the conservative mesh adaptation framework to be coupled with different remeshing procedures, geometrical parameterizations, structural models. Current limitations in the software implementation have restricted the focus of this work to inviscid flows using a moderate number of degrees of freedom for the flow variables (less than one million mesh nodes). Optimization for high performance computing environments would require the implementation of multigrid techniques, fully parallel unstructured mesh redistribution algorithms, and a viscous flow model for further aeronautical applications. These issues do not directly concern mesh adaptation, and have not prevented the successful application of the conservative mesh adaptation methodology to non-trivial three dimensional aeroelastic problems. In order to extend the range of application of the numerical technique, future developments can include the following topics:

- Geometry time-parameterization could be extended and tested in three dimensions by means of Bézier surfaces.
- Extension of the reduced order model for camber-morphing constant-section wings to the dynamic case could be investigated through a convergence analysis of the projected model in the time domain.
- A fully conservative and consistent fluid–structure interaction with an elastic structural model could be studied by investigating the analogies between the conservative solution transfer methodology for adaptive meshes employed in this work and the supermesh approach for conservative interpolation between non-matching interface discretizations introduced in [60, 3].

Appendix: Deflation procedure for singular matrix pairs

In section 5.3.1 it is shown that the definition of the state space vector

$$\mathbf{q} = \begin{bmatrix} \mathbf{u} \\ \mathbf{u}_p \end{bmatrix} \tag{A.1}$$

leads to a compact description of the generalized unsymmetrical eigenvalue problem

$$(\mathbf{A} - \lambda_i \mathbf{B})\,\hat{\mathbf{q}}_i = \mathbf{0} \tag{A.2}$$

This system is twelve times singular, since neither the six rigid modes nor their spatial derivatives have been constrained through boundary conditions. The geometric multiplicity of the null eigenvalues is only four, meaning that only the three rigid translation and the rigid rotation around the beam axis are eigenfunctions of the matrix pair. Since the system is not diagonalizable, each null eigenvalue is associated to a Jordan canonical form.

A.0.1 Deflation procedure

Since the problem is not diagonalizable, it is not possible to desingularize the problem by imposing the orthogonality of the solution with respect to the eigenfunctions. The knowledge of the Jordan canonical form associated with the null eigenvalues, however, allows to devise a deflation procedure to move the singularity away from the origin, without affecting the rest of the spectrum and the shape of the eigenfunctions.

To this aim, we introduce the complete right and left Jordan canonical forms of matrix pair $(\mathbf{A}, \mathbf{B})$

$$\mathbf{AX} = \mathbf{BXJ} \qquad \mathbf{Y}^H\mathbf{A} = \mathbf{JY}^H\mathbf{B} \tag{A.3}$$

with the normalization condition

$$\mathbf{Y}^H\mathbf{BX} = \mathbf{I} \tag{A.4}$$

The introduction of the left Jordan vectors allows to decompose matrix $\mathbf{A}$ into the Jordan canonical form

$$\mathbf{A} = \mathbf{BXJY}^H\mathbf{B} \tag{A.5}$$

without the need for an explicit inverse of the right vectors matrix $\mathbf{X}$. This decomposition shows that it is possible to modify the spectrum of matrix $\mathbf{A}$, without affecting its Jordan vectors[1], simply by modifying its Jordan matrix $\mathbf{J}$. So, a block Hotelling's deflation procedure [142, 27] for matrix $\mathbf{A}$ can be devised, through the addition of a simple diagonal shift $\mathbf{\Sigma}$ on the singular Jordan blocks

$$\tilde{\mathbf{A}} = \mathbf{B}\mathbf{X}(\mathbf{J}+\mathbf{\Sigma})\mathbf{Y}^H\mathbf{B} = \mathbf{A}+\mathbf{B}\mathbf{X}\mathbf{\Sigma}\mathbf{Y}^H\mathbf{B} \tag{A.6}$$

Since we want to modify only the null eigenvalues, the shift $\mathbf{\Sigma}$ assumes the expression

$$\mathbf{\Sigma} = \begin{bmatrix} \mathbf{\Sigma}_0 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}, \qquad \mathbf{\Sigma}_0 = \mathrm{diag}(\sigma_1, \ldots, \sigma_{12}) \tag{A.7}$$

with equal shifts on eigenvalues related to the same Jordan block. Thus, the deflated matrix reads

$$\tilde{\mathbf{A}} = \mathbf{A}+\mathbf{B}\mathbf{X}_0\mathbf{\Sigma}_0\mathbf{Y}_0^H\mathbf{B} \tag{A.8}$$

with

$$\mathbf{X}_0 = \begin{bmatrix}\mathbf{x}_1, \ldots, \mathbf{x}_{12}\end{bmatrix}, \quad \mathbf{Y}_0 = \begin{bmatrix}\mathbf{y}_1, \ldots, \mathbf{y}_{12}\end{bmatrix} \tag{A.9}$$

In this way, just the knowledge of the right and left Jordan vectors associated to the zero eigenvalues is required.

The new matrix pair $(\tilde{\mathbf{A}}, \mathbf{B})$ possesses the same spectrum of the original matrix pair $(\mathbf{A}, \mathbf{B})$, except from the zero eigenvalues which are shifted according to the non-null diagonal values of $\mathbf{\Sigma}_0$. The right and left eigenvectors of matrix $\tilde{\mathbf{A}}$ are the same of matrix $\mathbf{A}$, as can be shown thanks to the normalization condition A.4.

In fact, by right-multiplying the deflated matrix by a right principal vector $\mathbf{x}_i$ of the matrix pair $(\mathbf{A}, \mathbf{B})$, such that $\mathbf{A}\mathbf{x}_i = \lambda_i\mathbf{B}\mathbf{x}_i + k\mathbf{B}\mathbf{x}_{i-1}$ (with $k=0$ if $\mathbf{x}_i$ is an eigenvector, $k=1$ if $\mathbf{x}_i$ is a Jordan vector) and exploiting the normalization condition, we get

$$\begin{aligned} \tilde{\mathbf{A}}\mathbf{x}_i &= \mathbf{A}\mathbf{x}_i + \mathbf{B}\mathbf{X}_0\mathbf{\Sigma}_0\mathbf{Y}_0^H\mathbf{B}\mathbf{x}_i = \\ &= \boldsymbol{\lambda}_i\mathbf{B}\mathbf{x}_i + k\mathbf{B}\mathbf{x}_{i-1} + \mathbf{B}\mathbf{X}_0\mathbf{\Sigma}_0\mathbf{Y}_0^H\mathbf{B}\mathbf{x}_i = \\ &= \begin{cases} (\lambda_i+\sigma_i)\mathbf{B}\mathbf{x}_i + k\mathbf{B}\mathbf{x}_{i-1}, & i = 1, \ldots, 12 \\ \lambda_i\mathbf{B}\mathbf{x}_i + k\mathbf{B}\mathbf{x}_{i-1}, & i > 12 \end{cases} \end{aligned} \tag{A.10}$$

Similarly, by left-multiplying the deflated matrix by a right eigenvector $\mathbf{y}_i$ of the matrix pair $(\mathbf{A}, \mathbf{B})$, such that $\mathbf{y}_i^H\mathbf{A} = \mathbf{y}_i^H\lambda_i\mathbf{B} + k\mathbf{y}_{i+1}^H\mathbf{B}$ (with $k=0$ if $\mathbf{y}_i$ is an eigenvector, $k=1$ if $\mathbf{y}_i$ is a Jordan vector), we get

$$\begin{aligned} \mathbf{y}_i^H\tilde{\mathbf{A}} &= \mathbf{y}_i^H\mathbf{A} + \mathbf{y}_i^H\mathbf{B}\mathbf{X}_0\mathbf{\Sigma}_0\mathbf{Y}_0^H\mathbf{B} = \\ &= \boldsymbol{\lambda}_i\mathbf{y}_i^H\mathbf{B} + k\mathbf{y}_{i+1}^H\mathbf{B} + \mathbf{y}_i^H\mathbf{B}\mathbf{X}_0\mathbf{\Sigma}_0\mathbf{Y}_0^H\mathbf{B} = \\ &= \begin{cases} (\lambda_i+\sigma_i)\mathbf{y}_i^H\mathbf{B} + k\mathbf{y}_{i+1}^H\mathbf{B}, & i = 1, \ldots, 12 \\ \lambda_i\mathbf{y}_i^H\mathbf{B} + k\mathbf{y}_{i+1}^H\mathbf{B}, & i > 12 \end{cases} \end{aligned} \tag{A.11}$$

[1] In case of non defective eigenvalues, Jordan vectors simply coincide with eigenvectors.

In the next section, a procedure for the computation of the right and left Jordan vectors is shown.

A.0.2 Computation of the right and left Jordan vectors associated to null eigenvalues

In order to compute the right and left Jordan vectors of system 5.24, the same procedure introduced in [125] is here extended to the computation of left Jordan vectors. Exploiting the block structure of matrices $\mathbf{A}, \mathbf{B}$, the direct and adjoint systems read[2]

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{bmatrix} \frac{\mathrm{d}}{\mathrm{d}x} \begin{bmatrix} \mathbf{u} \\ \mathbf{u}_p \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{E} & \mathbf{H} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{u}_p \end{bmatrix} \tag{A.12}$$

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{bmatrix} \frac{\mathrm{d}}{\mathrm{d}x} \begin{bmatrix} \mathbf{v} \\ \mathbf{v}_p \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{E} \\ \mathbf{I} & -\mathbf{H} \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ \mathbf{v}_p \end{bmatrix} \tag{A.13}$$

Explicit expressions for the blocks of the state vectors can be obtained as functions of the state derivatives

$$\begin{cases} \mathbf{E}\mathbf{u} = -\mathbf{H}\mathbf{u}' + \mathbf{M}\mathbf{u}'' \\ \mathbf{u}_p = \mathbf{u}' \end{cases} \tag{A.14}$$

$$\begin{cases} \mathbf{E}\mathbf{v}_p = \mathbf{H}\mathbf{v}_p' + \mathbf{M}\mathbf{v}_p'' \\ \mathbf{v} = \mathbf{H}\mathbf{v}_p + \mathbf{M}\mathbf{v}_p' \end{cases} \tag{A.15}$$

Starting from the eigenvectors as generators of the Jordan chains, right and left Jordan vectors are obtained from the previous relations by substituting first and second derivatives with the vectors obtained at the first and second previous degrees of the chains computation, when these degrees are available

$$\begin{cases} \mathbf{E}\mathbf{u}^{(i)} = -\mathbf{H}\mathbf{u}^{(i-1)} + \mathbf{M}\mathbf{u}^{(i-2)} \\ \mathbf{u}_p^{(i)} = \mathbf{u}^{(i-1)} \end{cases} \tag{A.16}$$

$$\begin{cases} \mathbf{E}\mathbf{v}_p^{(i)} = \mathbf{H}\mathbf{v}_p^{(i+1)} + \mathbf{M}\mathbf{v}_p^{(i+2)} \\ \mathbf{v}^{(i)} = \mathbf{H}\mathbf{v}_p^{(i)} + \mathbf{M}\mathbf{v}_p^{(i+1)} \end{cases} \tag{A.17}$$

Left chains are computed backwards, since they are associated with the transpose of the Jordan form.

Both the computations require the numerical inversion of the symmetric matrix $\mathbf{E}$, which is four times singular. Its nullspace is formed by the three rigid translation and the rigid rotation around the beam axis, so it can be determined analytically. Unique solutions to the linear systems can be computed by imposing the

[2] For the problem at hand, continuos adjoint and discrete adjoint of system 5.24 coincide, and the derivation of the adjoint system (see [104]) is performed by simple transposition of the system matrices, exploiting the symmetry of matrices $\mathbf{E}$ and $\mathbf{M}$, and the skew-symmetry of matrix $\mathbf{H} = -\mathbf{H}^T$.

orthogonality of the solution to the rigid eigenfunctions $\mathbf{U}_0(\mathbf{x}) = \left[\mathbf{u}^{(1)}(\mathbf{x}) \dots \mathbf{u}^{(4)}(\mathbf{x})\right]$. Since the same procedure is applicable to both $\mathbf{u}$ and $\mathbf{v}$, we will refer here to a generic state $\mathbf{w}$ and a generic forcing term $\mathbf{f}$. The orthogonality condition for the continuous field $\mathbf{w}(\mathbf{x})$ reads

$$\langle \mathbf{U}_0, \mathbf{w} \rangle_{L^2(\mathscr{A})} = \mathbf{0} \tag{A.18}$$

After the discretization of the beam section (denoting fields and shape function coefficients by the same symbols), the orthogonality condition becomes

$$\mathbf{\Phi w} = \mathbf{0}, \qquad \mathbf{\Phi} = \mathbf{U}_0^T \int_{\mathscr{A}} \mathbf{N}^T(x^2, x^3)\mathbf{N}(x^2, x^3)\, \mathrm{d}\mathscr{A} \tag{A.19}$$

Using a vector of Lagrange multipliers $\boldsymbol{\lambda} \in \mathbb{R}^4$, the orthogonality is imposed by solving the extended linear systems

$$\begin{bmatrix} \mathbf{E} & \mathbf{\Phi}^T \\ \mathbf{\Phi} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \mathbf{0} \end{bmatrix} \tag{A.20}$$

Notice that a non null value of the Lagrange multipliers $\boldsymbol{\lambda}$ is produced when the compatibility condition requiring the orthogonality between the forcing $\mathbf{f}$ and the kernel of the adjoint system is violated, suggesting a method to detect when the maximum degree of a Jordan block is reached during the numerical computation of Jordan vectors. Thanks to the symmetry of $\mathbf{E}$, the same procedure is used to orthogonalize the adjoint variable $\mathbf{v}_p$ with respect to the rigid modes.

Using this orthogonalization procedure, right and left Jordan vectors are computed until the maximum degree of each Jordan block is reached. Since the problem is derogatory[3], an arbitrary linear combination of principal vectors *from different Jordan blocks* also satisfies the Jordan chains [74]. Thus, at each degree it is necessary to compute the Jordan vectors in all the blocks and epurate them from the contribution given by Jordan vectors of already completed blocks, which can be recognized by a non-null value of the Lagrange multipliers $\tilde{\boldsymbol{\lambda}}$. Thus, at each degree i, vectors in block k are updated eliminating contributions from the completed block j through a Gram-Schmidt orthogonalization procedure. As before, the same procedure applies to the computation of left Jordan chains.

[3] An eigenvalue problem is called derogatory when it is defective and multiple Jordan blocks are associated to the same eigenvalue [74].

Bibliography

[1] R. Abgrall, H. Beaugendre, and C. Dobrzynski. “An immersed boundary method using unstructured anisotropic mesh adaptation combined with level-sets and penalization techniques”. In: *Journal of Computational Physics* 257 (2014), pp. 83–101 (cit. on p. 23).

[2] M. Aftosmis and N. Kroll. “A quadrilateral based second-order TVD method for unstructured adaptive meshes”. In: *Aerospace Sciences Meetings*. American Institute of Aeronautics and Astronautics, Jan. 1991, (cit. on pp. 32, 55).

[3] H. J. Aguerre, S. M. Damián, J. M. Gimenez, and N. M. Nigro. “Conservative handling of arbitrary non-conformal interfaces using an efficient supermesh”. In: *Journal of Computational Physics* 335 (2017), pp. 21–49 (cit. on pp. 26, 149, 185).

[4] A. Airoldi, M. Crespi, G. Quaranta, and G. Sala. “Design of a Morphing Airfoil with Composite Chiral Structure”. In: *Journal of Aircraft* 49.4 (July 2012), pp. 1008–1019 (cit. on p. 116).

[5] F. Alauzet, P. Frey, P. George, and B. Mohammadi. “3D transient fixed point mesh adaptation for time-dependent problems: Application to CFD simulations”. In: *Journal of Computational Physics* 222.2 (2007), pp. 592–623 (cit. on pp. 17, 25).

[6] F. Alauzet and M. Mehrenberger. “P1-conservative solution interpolation on unstructured triangular meshes”. In: *International Journal for Numerical Methods in Engineering* 84.13 (2010), pp. 1552–1588 (cit. on p. 24).

[7] F. Alauzet. “A parallel matrix-free conservative solution interpolation on unstructured tetrahedral meshes”. In: *Computer Methods in Applied Mechanics and Engineering* 299 (2016), pp. 116–142 (cit. on pp. 18, 23, 25, 26).

[8] F. Alauzet and P. Frey. *Estimateur d’erreur géométrique et métriques anisotropes pour l’adaptation de maillage. Partie I : aspects théoriques*. Research Report RR-4759. INRIA, 2003 (cit. on pp. 30, 31, 55).

[9] F. Alauzet and A. Loseille. “A decade of progress on anisotropic mesh adaptation for computational fluid dynamics”. In: *Computer-Aided Design* 72 (2016). 23rd International Meshing Roundtable Special Issue: Advances in Mesh Generation, pp. 13–39 (cit. on pp. 17, 22).

[10] M. Alexa. “Recent advances in mesh morphing”. In: *Computer graphics forum*. Vol. 21. 2. Wiley Online Library. 2002, pp. 173–198 (cit. on p. 108).

[11] J. Anderson. *Modern Compressible Flow: With Historical Perspective*. Aeronautical and Aerospace Engineering Series. McGraw-Hill Education, 2003 (cit. on p. 18).

[12] A. Antoulas. *Approximation of Large-Scale Dynamical Systems*. Advances in Design and Control. Society for Industrial and Applied Mathematics, 2009 (cit. on pp. 117, 121, 124, 127).

[13] L. Arpaia and M. Ricchiuto. *Mesh adaptation by continuous deformation. Basics: accuracy, efficiency, well balancedness*. Research Report RR-8666. Inria Bordeaux Sud-Ouest ; INRIA, Jan. 2015 (cit. on p. 29).

[14] L. Arpaia and M. Ricchiuto. *r-adaptation for Shallow Water flows: conservation, well balancedness, efficiency*. Research Report RR-8956. Inria Bordeaux Sud-Ouest, 2016 (cit. on p. 29).

[15] S. Barbarino, O. Bilgen, R. Ajaj, M. Friswell, and D. Inman. “A Review of Morphing Aircraft”. In: *Journal of Intelligent Material Systems and Structures* 22.9 (June 2011), pp. 823–877 (cit. on pp. 115, 116).

[16] N. Barral, G. Olivier, and F. Alauzet. “Time-accurate anisotropic mesh adaptation for three-dimensional time-dependent problems with body-fitted moving geometries”. In: *Journal of Computational Physics* 331 (Feb. 2017), pp. 157–187 (cit. on p. 18).

[17] J. T. Batina. “Unsteady Euler airfoil solutions using unstructured dynamic meshes”. In: *AIAA journal* 28.8 (1990), pp. 1381–1388 (cit. on p. 46).

[18] O. A. Bauchau and S. Han. *Advanced Beam Theory for Multibody Dynamics*. 2013 (cit. on p. 122).

[19] O. A. Bauchau and S. Han. “Three-Dimensional Beam Theory for Flexible Multibody Dynamics”. In: *Journal of Computational and Nonlinear Dynamics* 9.4 (July 2014) (cit. on p. 122).

[20] T. Belytschko, Y. Krongauz, D. Organ, M. Fleming, and P. Krysl. “Meshless methods: An overview and recent developments”. In: *Computer Methods in Applied Mechanics and Engineering* 139.1 (1996), pp. 3–47 (cit. on pp. 149, 150).

[21] O. O. Bendiksen. "Modern developments in computational aeroelasticity". In: *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering* 218.3 (2004), pp. 157–177 (cit. on p. 17).

[22] O. O. Bendiksen. "Nonclassical aileron buzz in transonic flow". In: *Structures, Structural Dynamics, and Materials and Co-located Conferences*. American Institute of Aeronautics and Astronautics, Apr. 1993 (cit. on pp. 158–160, 169).

[23] O. O. Bendiksen. "Review of unsteady transonic aerodynamics: Theory and applications". In: *Progress in Aerospace Sciences* 47.2 (2011), pp. 135–167 (cit. on pp. 157, 159, 160, 183).

[24] P. Benner and H. FaSSbender. "An implicitly restarted symplectic Lanczos method for the Hamiltonian eigenvalue problem". In: *Linear Algebra and its Applications* 263 (1997), pp. 75–111 (cit. on p. 125).

[25] P. Benner, V. Mehrmann, and H. Xu. "A numerically stable, structure preserving method for computing the eigenvalues of real Hamiltonian or symplectic pencils". In: *Numerische Mathematik* 78.3 (Jan. 1998), pp. 329–358 (cit. on p. 125).

[26] R. Bisplinghoff, H. Ashley, and R. Halfman. *Aeroelasticity*. Dover Books on Aeronautical Engineering. Dover Publications, 2013 (cit. on p. 115).

[27] E. Bodewig. *Matrix Calculus*. North Holland, 1959 (cit. on pp. 125, 188).

[28] D. Boffi and L. Gastaldi. "Stability and geometric conservation laws for ALE formulations". In: *Computer methods in applied mechanics and engineering* 193.42-44 (2004), pp. 4717–4739 (cit. on p. 21).

[29] D. L. Brown, W. D. Henshaw, and D. J. Quinlan. "Overture: An Object-Oriented Framework for Solving Partial Differential Equations". In: *Proceedings of the Scientific Computing in Object-Oriented Parallel Environments*. ISCOPE '97. London, UK, UK: Springer-Verlag, 1997, pp. 177–184 (cit. on p. 22).

[30] H. H. Brown, J. Rathert George A., and L. A. Clousing. *Flight-test measurements of aileron control surface behavior at super critical Mach numbers*. NACA Research Memorandum A7A15. Ames Aeronautical Laboratory, 1947 (cit. on pp. 160, 161, 164, 165, 173, 183).

[31] C. J. Budd, W. Huang, and R. D. Russell. "Adaptivity with moving grids". In: *Acta Numerica* 18 (2009), pp. 111–241 (cit. on p. 28).

[32] C. Budd and M. D. Piggott. "Geometric Integration and Its Applications". In: *in Handbook of numerical analysis*. North-Holland, 2000, pp. 35–139 (cit. on p. 30).

[33] A. Carbonara. “Sviluppo di modelli aerodinamici di ordine ridotto per lanalisi del buzz di alettone”. Politecnico di Milano, 2016 (cit. on pp. 105, 157, 161).

[34] G. Carpentieri. “An adjoint-based shape-optimization method for aerodynamic design”. PhD thesis. TU Delft, 2009 (cit. on pp. 45, 46).

[35] L. Cavagna, G. Quaranta, and P. Mantegazza. “Application of NavierStokes simulations for aeroelastic stability assessment in transonic regime”. In: *Computers & Structures* 85.11 (2007). Fourth MIT Conference on Computational Fluid and Solid Mechanics, pp. 818–832 (cit. on p. 105).

[36] J. R. Cebral and R. Lohner. “Conservative Load Projection and Tracking for Fluid-Structure Problems”. In: *AIAA Journal* 35.4 (Apr. 1997), pp. 687–692 (cit. on p. 147).

[37] H. D. Ceniceros and T. Y. Hou. “An Efficient Dynamically Adaptive Mesh for Potentially Singular Solutions”. In: *Journal of Computational Physics* 172.2 (2001), pp. 609–639 (cit. on pp. 29, 59, 60).

[38] G. Chen, H. Tang, and P. Zhang. “Second-order accurate Godunov scheme for multicomponent flows on moving triangular meshes”. In: *Journal of Scientific Computing* 34.1 (2008), pp. 64–86 (cit. on pp. 29, 59, 67, 74, 184).

[39] S. E. Chen and R. E. Parent. “Shape averaging and its applications to industrial design”. In: *IEEE Computer Graphics and Applications* 9.1 (Jan. 1989), pp. 47–54 (cit. on p. 108).

[40] X.-Y. Chen and G.-C. Zha. “Fully coupled fluidstructural interactions using an efficient high resolution upwind scheme”. In: *Journal of Fluids and Structures* 20.8 (2005), pp. 1105–1125 (cit. on p. 105).

[41] B. Cockburn. “Discontinuous Galerkin methods for convection-dominated problems”. In: *High-order methods for computational physics*. Springer, 1999, pp. 69–224 (cit. on p. 74).

[42] C. Dapogny, C. Dobrzynski, and P. Frey. “Three-dimensional adaptive domain remeshing, implicit domain meshing, and applications to free and moving boundary problems”. In: *Journal of Computational Physics* 262 (2014), pp. 358–378 (cit. on pp. 37, 52, 53, 183).

[43] C. Dapogny, C. Dobrzynski, P. Frey, and A. Froehly. *Mmg platform*. URL: `www.mmgtools.org` (cit. on p. 52).

[44] A. de Boer, A. van Zuijlen, and H. Bijl. “Comparison of conservative and consistent approaches for the coupling of non-matching meshes”. In: *Computer Methods in Applied Mechanics and Engineering* 197.49 (2008), pp. 4284–4297 (cit. on pp. 147, 149).

[45] C. Dobrzynski and P. Frey. "Anisotropic Delaunay Mesh Adaptation for Unsteady Simulations". In: *Proceedings of the 17th International Meshing Roundtable*. Ed. by R. V. Garimella. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 177–194 (cit. on pp. 37, 52, 183).

[46] J. Donea, A. Huerta, J.-P. Ponthot, and A. Rodríguez-Ferran. "Arbitrary LagrangianEulerian Methods". In: *Encyclopedia of Computational Mechanics*. John Wiley & Sons, Ltd, 2004 (cit. on pp. 18, 31, 37).

[47] A. Druz', N. Polyakov, and Y. Ustinov. "Homogeneous solutions and saint-venant problems for a naturally twisted rod". In: *Journal of Applied Mathematics and Mechanics* 60.4 (1996), pp. 657–664 (cit. on p. 123).

[48] A. Druz' and Y. Ustinov. "Green's tensor for an elastic cylinder and its applications in the development of the Saint-Venant theory". In: *Journal of Applied Mathematics and Mechanics* 60.1 (1996), pp. 97–104 (cit. on p. 123).

[49] G.-R. Duan. *Analysis and design of descriptor linear systems*. Vol. 23. Springer Science & Business Media, 2010 (cit. on p. 105).

[50] J. K. Dukowicz. "Conservative Rezoning (Remapping) for General Quadrilateral Meshes". In: *Journal of Computational Physics* 54 (June 1984), pp. 411–424 (cit. on p. 26).

[51] A. S. Dvinsky. "Adaptive grid generation from harmonic maps on Riemannian manifolds". In: *Journal of Computational Physics* 95.2 (1991), pp. 450–476 (cit. on pp. 29, 31).

[52] A. F. Emery. "An evaluation of several differencing methods for inviscid fluid flow problems". In: *Journal of Computational Physics* 2.3 (1968), pp. 306–331 (cit. on p. 74).

[53] A. L. Erickson and R. L. Mannes. *Wind-Tunnel Investigation of Transonic Aileron Flutter*. National Advisory Committee for Aeronautics, 1949 (cit. on p. 158).

[54] A. L. Erickson and J. D. Stephenson. *A Suggested Method of Analyzing for Transonic Flutter of Control Surfaces Based on Available Experimental Evidence*. National Advisory Committee for Aeronautics, 1947 (cit. on pp. 158, 161, 173).

[55] S. Étienne, A. Garon, and D. Pelletier. "Perspective on the geometric conservation law and finite element methods for ALE simulations of incompressible flow". In: *Journal of Computational Physics* 228.7 (2009), pp. 2313–2333 (cit. on pp. 20, 21).

[56] C. Farhat and M. Lesoinne. “Two efficient staggered algorithms for the serial and parallel solution of three-dimensional nonlinear transient aeroelastic problems”. In: *Computer Methods in Applied Mechanics and Engineering* 182.3 (2000), pp. 499–515 (cit. on pp. 103–105).

[57] C. Farhat, P. Geuzaine, and C. Grandmont. “The Discrete Geometric Conservation Law and the Nonlinear Stability of ALE Schemes for the Solution of Flow Problems on Moving Grids”. In: *Journal of Computational Physics* 174.2 (2001), pp. 669–694 (cit. on pp. 20, 21, 104).

[58] C. Farhat, M. Lesoinne, and P. Le Tallec. “Load and motion transfer algorithms for fluid/structure interaction problems with non-matching discrete interfaces: Momentum and energy conservation, optimal discretization and application to aeroelasticity”. In: *Computer methods in applied mechanics and engineering* 157.1-2 (1998), pp. 95–114 (cit. on pp. 147, 149).

[59] P. Farrell and J. Maddison. “Conservative interpolation between volume meshes by local Galerkin projection”. In: *Computer Methods in Applied Mechanics and Engineering* 200.1 (2011), pp. 89–100 (cit. on pp. 25, 26).

[60] P. Farrell, M. Piggott, C. Pain, G. Gorman, and C. Wilson. “Conservative interpolation between unstructured meshes via supermesh construction”. In: *Computer Methods in Applied Mechanics and Engineering* 198.33 (2009), pp. 2632–2642 (cit. on pp. 26, 149, 185).

[61] K. J. Fidkowski and D. L. Darmofal. “Review of Output-Based Error Estimation and Mesh Adaptation in Computational Fluid Dynamics”. In: *AIAA Journal* 49.4 (Apr. 2011), pp. 673–694 (cit. on p. 22).

[62] J. Fincham and M. Friswell. “Aerodynamic optimisation of a camber morphing aerofoil”. In: *Aerospace Science and technology* 43 (2015), pp. 245–255 (cit. on p. 132).

[63] L. Formaggia. “Data structures for unstructured mesh generation”. In: *Handbook of Grid Generation*. CRC Press, 1999. Chap. 14 (cit. on pp. 28, 56, 57).

[64] M. Fossati, A. Guardone, and L. Vigevano. “A nodepair finite element/volume mesh adaptation technique for compressible flows based on a hierarchical approach”. In: *International Journal for Numerical Methods in Fluids* 70.8 (2012), pp. 1004–1026 (cit. on p. 38).

[65] P. J. Frey and P.-L. George. *Mesh Generation: Application to Finite Elements*. ISTE, 2007 (cit. on pp. 23, 24, 31, 53, 56, 57, 151).

[66] P. Frey and F. Alauzet. “Anisotropic mesh adaptation for CFD computations”. In: *Computer Methods in Applied Mechanics and Engineering* 194.48 (2005). Unstructured Mesh Generation, pp. 5068–5082 (cit. on pp. 30, 31, 55, 66).

[67] F. Gandhi, M. Frecker, and A. Nissly. “Design Optimization of a Controllable Camber Rotor Airfoil”. In: *AIAA Journal* 46.1 (Jan. 2008), pp. 142–153 (cit. on p. 116).

[68] R. V. Garimella. “Mesh data structure selection for mesh generation and FEA applications”. In: *International Journal for Numerical Methods in Engineering* 55.4 (), pp. 451–478 (cit. on pp. 56, 57).

[69] C. Geuzaine, B. Meys, F. Henrotte, P. Dular, and W. Legros. “A Galerkin projection method for mixed finite elements”. In: *IEEE Transactions on Magnetics* 35.3 (May 1999), pp. 1438–1441 (cit. on p. 25).

[70] C. Geuzaine and J.-F. Remacle. “Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities”. In: *International journal for numerical methods in engineering* 79.11 (2009), pp. 1309–1331 (cit. on p. 162).

[71] P. Geuzaine, C. Grandmont, and C. Farhat. “Design and analysis of ALE schemes with provable second–order time–accuracy for inviscid and viscous flow simulations”. In: *Journal of Computational Physics* 191.1 (2003), pp. 206–227 (cit. on p. 20).

[72] V. Giavotto, M. Borri, P. Mantegazza, G. Ghiringhelli, V. Carmaschi, G. C. Maffioli, and F. Mussi. “Anisotropic beam theory and applications”. In: *Computers and Structures* 16 (1983) (cit. on pp. 116, 117).

[73] M. B. Giles. “Stability and accuracy of numerical boundary conditions in aeroelastic analysis”. In: *International Journal for Numerical Methods in Fluids* 24.8 (1998), pp. 739–757 (cit. on p. 105).

[74] G. H. Golub and J. H. Wilkinson. “Ill-conditioned eigensystems and the computation of the Jordan canonical form”. In: *SIAM Review* 18 (1976) (cit. on pp. 123, 124, 190).

[75] A. Guardone, D. Isola, and G. Quaranta. “Arbitrary Lagrangian Eulerian formulation for two-dimensional flows using dynamic meshes with edge swapping”. In: *Journal of Computational Physics* 230.20 (2011), pp. 7706–7722 (cit. on pp. 27, 37, 50, 183).

[76] A. Guardone and L. Vigevano. “Roe Linearization for the van der Waals Gas”. In: *Journal of Computational Physics* 175.1 (2002), pp. 50–78 (cit. on p. 42).

[77] A. Guardone and L. Quartapelle. *Unstructured finite–volume high–resolution methods for conservation laws*. Tech. rep. Dipartimento di Ingegneria Aerospaziale, Politecnico di Milano, 2000 (cit. on pp. 43, 44).

[78] H. Guillard and C. Farhat. "On the significance of the geometric conservation law for flow computations on moving meshes". In: *Computer Methods in Applied Mechanics and Engineering* 190.11 (2000), pp. 1467–1482 (cit. on p. 20).

[79] G. P. Guruswamy. "A review of numerical fluids/structures interface methods for computations using high-fidelity equations". In: *Computers & Structures* 80.1 (2002), pp. 31–41 (cit. on pp. 142, 146).

[80] M. W. Heinstein and T. A. Laursen. "A three dimensional surface-to-surface projection algorithm for non-coincident domains". In: *Communications in Numerical Methods in Engineering* 19.6 (), pp. 421–432 (cit. on p. 146).

[81] W. D. Henshaw. "Adaptive Mesh and Overlapping Grid Methods". In: *Encyclopedia of Aerospace Engineering*. John Wiley & Sons, Ltd, 2010 (cit. on p. 22).

[82] M. W. Hirsch, S. Smale, and R. L. Devaney. *Differential equations, dynamical systems, and an introduction to chaos*. Academic press, 2012 (cit. on p. 169).

[83] D. H. Hodges. *Nonlinear Composite Beam Theory*. American Institute of Aeronautics and Astronautics, 2006 (cit. on p. 116).

[84] C. O. Horgan. "Recent Developments Concerning Saint-Venants Principle: A Second Update". In: *Applied Mechanics Reviews* 49.10S (Oct. 1996), S101–S111 (cit. on p. 117).

[85] C. O. Horgan. "Recent Developments Concerning Saint-Venants Principle: An Update". In: *Applied Mechanics Reviews* 42.11 (Nov. 1989), pp. 295–303 (cit. on p. 117).

[86] J. T. Howlett. *Calculation of unsteady transonic flows with mild separation by viscous-inviscid interaction*. NASA Technical Paper. NASA Langley Research Center, 1992 (cit. on pp. 164, 173).

[87] W. Huang. "Variational Mesh Adaptation: Isotropy and Equidistribution". In: *Journal of Computational Physics* 174.2 (2001), pp. 903–924 (cit. on p. 29).

[88] W. Huang and L. Kamenski. "A geometric discretization and a simple implementation for variational mesh generation and adaptation". In: *Journal of Computational Physics* 301 (2015), pp. 322–337 (cit. on p. 30).

[89] W. Huang and L. Kamenski. “On the mesh nonsingularity of the moving mesh PDE method”. In: *Mathematics of Computation* 87 (2018), pp. 1887–1911 (cit. on p. 30).

[90] W. Huang and R. D. Russell. “Adaptive mesh movement the MMPDE approach and its applications”. In: *Journal of Computational and Applied Mathematics* 128.1 (2001). Numerical Analysis 2000. Vol. VII: Partial Differential Equations, pp. 383–398 (cit. on p. 29).

[91] W. Huang and W. Sun. “Variational mesh adaptation II: error estimates and monitor functions”. In: *Journal of Computational Physics* 184.2 (2003), pp. 619–648 (cit. on p. 29).

[92] D. Ibanez, N. Barral, J. Krakos, A. Loseille, T. Michal, and M. Park. “First benchmark of the Unstructured Grid Adaptation Working Group”. In: *Procedia Engineering* 203 (2017). 26th International Meshing Roundtable, IMR26, 18-21 September 2017, Barcelona, Spain, pp. 154–166 (cit. on p. 17).

[93] D. Isola. “An interpolation-free two-dimensional conservative ALE scheme over adaptive unstructured grids for rotorcraft aerodynamics”. PhD thesis. Politecnico di Milano, 2012 (cit. on pp. 41, 46, 54).

[94] D. Isola, A. Guardone, and G. Quaranta. “Finite-volume solution of two-dimensional compressible flows over dynamic adaptive grids”. In: *Journal of Computational Physics* 285 (2015), pp. 1–23 (cit. on pp. 18, 21, 25, 27, 30, 32, 37, 50, 183).

[95] A. Johnson and T. Tezduyar. “Mesh update strategies in parallel finite element computations of flow problems with moving boundaries and interfaces”. In: *Computer Methods in Applied Mechanics and Engineering* 119.1 (1994), pp. 73–94 (cit. on pp. 20, 30).

[96] A. Johnson and T. Tezduyar. “Simulation of multiple spheres falling in a liquid-filled tube”. In: *Computer Methods in Applied Mechanics and Engineering* 134.3 (1996), pp. 351–373 (cit. on p. 30).

[97] V. Kalro and T. E. Tezduyar. “A parallel 3D computational method for fluidstructure interactions in parachute systems”. In: *Computer Methods in Applied Mechanics and Engineering* 190.3 (2000), pp. 321–332 (cit. on p. 30).

[98] L. Kamenski and W. Huang. “How a Nonconvergent Recovered Hessian Works in Mesh Adaptation”. In: *SIAM Journal on Numerical Analysis* 52.4 (2014), pp. 1692–1708 (cit. on p. 22).

[99] B. Koren. “Defect correction and multigrid for an efficient and accurate computation of airfoil flows”. In: *Journal of Computational Physics* 77.1 (1988), pp. 183–206 (cit. on p. 45).

[100] M. Kucharik and M. Shashkov. “Extension of efficient, swept-integration-based conservative remapping method for meshes with changing connectivity”. In: *International Journal for Numerical Methods in Fluids* 56.8 (2008), pp. 1359–1365 (cit. on p. 26).

[101] S. K. Lahiri, J. Bonet, and J. Peraire. “A variationally consistent mesh adaptation method for triangular elements in explicit Lagrangian dynamics”. In: *International Journal for Numerical Methods in Engineering* 82.9 (2010), pp. 1073–1113 (cit. on p. 27).

[102] N. C. Lambourne. *Control–surface buzz*. Aeronautical Research Council Reports and Memoranda, 1962 (cit. on pp. 157–159).

[103] P. Lancaster and K. Salkauskas. “Surfaces generated by moving least squares methods”. In: *Mathematics of computation* 37.155 (1981), pp. 141–158 (cit. on p. 149).

[104] C. Lanczos. *Linear Differential Operators*. Martino Publishing, 2012 (Reprint of 1961 edition) (cit. on pp. 121, 127, 189).

[105] C. Lanczos. *The Variational Principles of Mechanics*. Dover Publications, 1970 (cit. on p. 31).

[106] T. A. Laursen and M. W. Heinstein. “Consistent mesh tying methods for topologically distinct discretized surfaces in nonlinear solid mechanics”. In: *International Journal for Numerical Methods in Engineering* 57.9 (2003), pp. 1197–1242 (cit. on p. 146).

[107] F. Lazarus and A. Verroust. “Three-dimensional metamorphosis: a survey”. In: *The Visual Computer* 14.8 (Dec. 1998), pp. 373–389 (cit. on p. 108).

[108] R. J. LeVeque. *Finite volume methods for hyperbolic problems*. Vol. 31. Cambridge university press, 2002 (cit. on p. 42).

[109] R. LeVeque. *Numerical Methods for Conservation Laws*. Lectures in Mathematics ETH Zürich, Department of Mathematics Research Institute of Mathematics. Springer, 1992 (cit. on pp. 18, 42).

[110] R. Li, T. Tang, and P. Zhang. “A Moving Mesh Finite Element Algorithm for Singular Problems in Two and Three Space Dimensions”. In: *Journal of Computational Physics* 177.2 (2002), pp. 365–393 (cit. on p. 30).

[111] S. Li and L. Petzold. “Moving Mesh Methods with Upwinding Schemes for Time-Dependent PDEs”. In: *Journal of Computational Physics* 131.2 (1997), pp. 368–377 (cit. on p. 29).

[112] C. V. Loan. “A symplectic method for approximating all the eigenvalues of a Hamiltonian matrix”. In: *Linear Algebra and its Applications* 61 (1984), pp. 233–251 (cit. on p. 125).

[113] R. Löhner. “An adaptive finite element scheme for transient problems in CFD”. In: *Computer Methods in Applied Mechanics and Engineering* 61.3 (1987), pp. 323–338 (cit. on p. 54).

[114] R. Löhner. “Robust, Vectorized Search Algorithms for Interpolation on Unstructured Grids”. In: *Journal of Computational Physics* 118.2 (1995), pp. 380–387 (cit. on pp. 24, 32).

[115] R. Ma, X. Chang, L. Zhang, X. He, and M. Li. “On the Geometric Conservation Law for Unsteady Flow Simulations on Moving Mesh”. In: *Procedia Engineering* 126 (2015). Frontiers in Fluid Mechanics Research, pp. 639–644 (cit. on p. 21).

[116] L. G. Margolin and M. Shashkov. “Second-order Sign-preserving Conservative Interpolation (Remapping) on General Grids”. In: *J. Comput. Phys.* 184.1 (Jan. 2003), pp. 266–298 (cit. on p. 26).

[117] D. J. Mavriplis and Z. Yang. “Construction of the discrete geometric conservation law for high-order time-accurate simulations on dynamic meshes”. In: *Journal of Computational Physics* 213.2 (2006), pp. 557–573 (cit. on pp. 20, 21).

[118] S. Menon and D. P. Schmidt. “Conservative interpolation on unstructured polyhedral meshes: An extension of the supermesh approach to cell-centered finite-volume variables”. In: *Computer Methods in Applied Mechanics and Engineering* 200.41 (2011), pp. 2797–2804 (cit. on pp. 26, 149).

[119] C. Michler, S. Hulshoff, E. van Brummelen, and R. de Borst. “A monolithic approach to fluidstructure interaction”. In: *Computers & Fluids* 33.5 (2004). Applied Mathematics for Industrial Flow Problems, pp. 839–848 (cit. on p. 105).

[120] A. Mielke. *Hamiltonian and Lagrangian Flows on Center Manifolds - with Applications to Elliptic Variational Problems*. Springer-Verlag, 1991 (cit. on pp. 117, 123).

[121] A. Mielke. “Saint-Venant’s problem and semi-inverse solutions in nonlinear elasticity”. In: *Archive for Rational Mechanics and Analysis* 102.3 (Sept. 1988), pp. 205–229 (cit. on p. 123).

[122] M. Mistry and F. Gandhi. “Design, fabrication, and benchtop testing of a helicopter rotor blade section with warp-induced spanwise camber variation”. In: *Journal of Intelligent Material Systems and Structures* (Aug. 2014), pp. 1–18 (cit. on p. 116).

[123] R. Mittal and G. Iaccarino. “IMMERSED BOUNDARY METHODS”. In: *Annual Review of Fluid Mechanics* 37.1 (2005), pp. 239–261 (cit. on p. 23).

[124] B. Mohammadi, P.-L. George, F. Hecht, and E. Saltel. “3D Mesh adaptation by metric control for CFD”. In: *Revue Européenne des Éléments Finis* 9.4 (2000), pp. 439–449 (cit. on p. 30).

[125] M. Morandini, M. Chierichetti, and P. Mantegazza. “Characteristic behavior of prismatic anisotropic beam via generalized eigenvectors”. In: *International Journal of Solids and Structures* 47 (2010) (cit. on pp. 117, 122–124, 126, 131, 184, 189).

[126] D. Muffo, G. Quaranta, A. Guardone, and P. Mantegazza. *Interface Velocity Consistency in time-accurate flow simulations on dynamic meshes*. Tech. rep. Dipartimento di Ingegneria Aerospaziale, Politecnico di Milano, 2007 (cit. on p. 41).

[127] F. Nobile and L. Formaggia. “A Stability Analysis for the Arbitrary Lagrangian : Eulerian Formulation with Finite Elements”. In: *East-West Journal of Numerical Mathematics* 7.2 (1999), pp. 105–132 (cit. on p. 21).

[128] L. Nouveau, H. Beaugendre, M. Ricchiuto, C. Dobrzynski, and R. Abgrall. *An adaptive ALE residual based penalization approach for laminar flows with moving bodies*. Research Report RR-8936. INRIA Bordeaux, équipe CARDAMOM, July 2016, p. 15 (cit. on pp. 30, 59).

[129] M. A. Park, A. Loseille, J. Krakos, T. R. Michal, and J. J. Alonso. “Unstructured Grid Adaptation: Status, Potential Impacts, and Recommended Investments Towards CFD 2030”. In: *AIAA AVIATION Forum*. American Institute of Aeronautics and Astronautics, June 2016, (cit. on pp. 17, 22, 30).

[130] H. Pearcey. *Some effects of shock-induced separation of turbulent boundary layers in transonic flow past aerofoils*. 1955 (cit. on p. 160).

[131] M. Pelanti, L. Quartapelle, and L. Vigevano. *A review of entropy fixes as applied to Roe’s linearization*. Tech. rep. Dipartimento di Ingegneria Aerospaziale, Politecnico di Milano, 2001 (cit. on p. 43).

[132] S. Piperno and C. Farhat. "Partitioned procedures for the transient solution of coupled aeroelastic problems Part II: energy transfer analysis and three-dimensional applications". In: *Computer Methods in Applied Mechanics and Engineering* 190.24 (2001). Advances in Computational Methods for Fluid-Structure Interaction, pp. 3147–3170 (cit. on pp. 103–105).

[133] S. Piperno, C. Farhat, and B. Larrouturou. "Partitioned procedures for the transient solution of coupled aroelastic problems Part I: Model problem, theory and two-dimensional application". In: *Computer Methods in Applied Mechanics and Engineering* 124.1 (1995), pp. 79–112 (cit. on pp. 103, 105).

[134] W. H. Press. *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press, 2007 (cit. on p. 46).

[135] J. S. Przemieniecki. *Theory of matrix structural analysis*. Courier Corporation, 1985 (cit. on pp. 60, 108, 116, 118).

[136] G. Quaranta, P. Masarati, and P. Mantegazza. "A conservative mesh-free approach for fluid-structure interface problems". In: *International Conference for Coupled Problems in Science and Engineering, Greece*. 2005 (cit. on pp. 146, 147, 149).

[137] L. Quartapelle and F. Auteri. *Fluidodinamica comprimibile*. CEA, 2013 (cit. on p. 40).

[138] A. Quarteroni. *Modellistica Numerica per Problemi Differenziali*. UNITEXT. Springer Milan, 2009 (cit. on pp. 31, 121).

[139] A. Quarteroni, R. Sacco, and F. Saleri. *Matematica numerica*. Springer, 2008 (cit. on pp. 44, 64, 65).

[140] B. Re, C. Dobrzynski, and A. Guardone. "An interpolation-free ALE scheme for unsteady inviscid flows computations with large boundary displacements over three-dimensional adaptive grids". In: *Journal of Computational Physics* 340 (2017), pp. 26–54 (cit. on pp. 18, 25, 27, 37, 50, 56, 183).

[141] B. Re. "An adaptive interpolation-free conservative scheme for the three-dimensional Euler equations on dynamic meshes for aeronautical applications". PhD thesis. Politecnico di Milano, 2016 (cit. on pp. 41, 52).

[142] Y. Saad. *Numerical Methods for Large Eigenvalue Problems*. Society for Industrial and Applied Mathematics, 2011 (cit. on pp. 125, 188).

[143] K. Saito, F. Agnese, and F. Scarpa. "A Cellular Kirigami Morphing Wingbox Concept". In: *Journal of Intelligent Material Systems and Structures* 22.9 (June 2011), pp. 935–944 (cit. on p. 116).

[144] V. Schmitt and F. Charpin. *Pressure Distributions on the ONERA–M6-Wing at Transonic Mach Numbers*. Experimental Data Base for Computer Program Assessment, AGARD AR 138. Report of the Fluid Dynamics Panel Working Group 04, 1979 (cit. on p. 57).

[145] V. Selmin. "The node-centred finite volume approach: Bridge between finite differences and finite elements". In: *Computer Methods in Applied Mechanics and Engineering* 102.1 (1993), pp. 107–138 (cit. on pp. 38, 43).

[146] V. Selmin and L. Formaggia. "Unified construction of finite element and finite volume discretizations for compressible flows". In: *International Journal for Numerical Methods in Engineering* 39.1 (1996), pp. 1–32 (cit. on p. 39).

[147] J. H. Seo and R. Mittal. "A sharp-interface immersed boundary method with improved mass conservation and reduced spurious pressure oscillations". In: *Journal of Computational Physics* 230.19 (2011), pp. 7347–7363 (cit. on p. 23).

[148] S. Skogestad and I. Postlethwaite. *Multivariable feedback control: analysis and design*. Vol. 2. Wiley New York, 2007 (cit. on pp. 121, 124).

[149] S. R. Slattery. "Mesh-free data transfer algorithms for partitioned multiphysics problems: Conservation, accuracy, and parallelism". In: *Journal of Computational Physics* 307 (2016), pp. 164–188 (cit. on p. 149).

[150] J. Slotnick, A. Khodadoust, J. Alonso, D. Darmofal, W. Gropp, E. Lurie, and D. Mavriplis. *CFD vision 2030 study: a path to revolutionary computational aerosciences*. Tech. rep. NASA CR-2014-218178, 2014 (cit. on p. 17).

[151] F. Sotiropoulos and X. Yang. "Immersed boundary methods for simulating fluidstructure interaction". In: *Progress in Aerospace Sciences* 65 (2014), pp. 1–21 (cit. on p. 23).

[152] K. Stein, T. Tezduyar, and R. Benney. "Mesh Moving Techniques for Fluid-Structure Interactions With Large Displacements". In: *Journal of Applied Mechanics* 70.1 (Jan. 2003), pp. 58–63 (cit. on p. 30).

[153] H. Tang and T. Tang. "Adaptive Mesh Methods for One- and Two-Dimensional Hyperbolic Conservation Laws". In: *SIAM Journal on Numerical Analysis* 41.2 (2003), pp. 487–515 (cit. on p. 29).

[154] T. Tang. "Moving mesh methods for computational fluid dynamics". In: *Contemporary mathematics* 383 (2005), pp. 141–174 (cit. on pp. 29, 67).

[155] G. Teschl. *Ordinary differential equations and dynamical systems*. Graduate studies in mathematics. Vol. 140. American Mathematical Society, 2012 (cit. on p. 169).

[156] P. Thomas and C. Lombard. "Geometric conservation law and its application to flow computations on moving grids". In: *AIAA journal* 17.10 (1979), pp. 1030–1037 (cit. on p. 20).

[157] J. Thompson, Z. Warsi, and C. Mastin. *Numerical grid generation: foundations and applications*. North-Holland, 1985 (cit. on pp. 28, 29, 31).

[158] H. Tijdeman. "Investigations of the transonic flow around oscillating airfoils". In: (1977) (cit. on pp. 94, 98, 157–159).

[159] S. Timoshenko and J. Goodier. *Theory of Elasticity*. McGraw-Hill Book Company, 1951 (cit. on p. 116).

[160] E. F. Toro. *Riemann solvers and numerical methods for fluid dynamics: a practical introduction*. Springer Science & Business Media, 2013 (cit. on p. 42).

[161] J. G. Trulio and K. R. Trigger. *Numerical solution of the one-dimensional Lagrangian hydrodynamic equations*. Tech. rep. California. Univ., Livermore, CA (United States). Lawrence Radiation Lab., 1961 (cit. on p. 20).

[162] B. van Leer. "Towards the ultimate conservative difference scheme. II. Monotonicity and conservation combined in a second-order scheme". In: *Journal of Computational Physics* 14.4 (1974), pp. 361–370 (cit. on pp. 42, 43).

[163] B. van Leer. "Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method". In: *Journal of Computational Physics* 32.1 (1979), pp. 101–136 (cit. on p. 74).

[164] V. Venkatakrishnan and D. Mavriplis. "Implicit Method for the Computation of Unsteady Flows on Unstructured Grids". In: *Journal of Computational Physics* 127.2 (1996), pp. 380–397 (cit. on p. 45).

[165] S. Völkner, J. Brunswig, and T. Rung. "Analysis of non-conservative interpolation techniques in overset grid finite-volume methods". In: *Computers & Fluids* 148 (2017), pp. 39–55 (cit. on pp. 23, 25).

[166] G. WARREN, W. ANDERSON, J. THOMAS, and S. KRIST. "Grid convergence for adaptive methods". In: *Fluid Dynamics and Co-located Conferences*. American Institute of Aeronautics and Astronautics, June 1991, (cit. on p. 22).

[167] B. E. Webster, M. S. Shephard, Z. Rusak, and J. E. Flaherty. "Automated adaptive time-discontinuous finite element method for unsteady compressible airfoil aerodynamics". In: *AIAA Journal* 32.4 (Apr. 1994), pp. 748–757 (cit. on pp. 30, 32, 54).

[168] T. Weisshaar. “Morphing Aircraft Systems: Historical Perspectives and Future Challenges”. In: *Journal of Aircraft* 50.2 (Mar. 2013), pp. 337–353 (cit. on p. 116).

[169] A. M. Winslow. *Adaptive-mesh zoning by the equipotential method*. Tech. rep. Lawrence Livermore National Lab., CA (USA), 1981 (cit. on p. 29).

[170] B. Wohlmuth. “Variationally consistent discretization schemes and numerical algorithms for contact problems”. In: *Acta Numerica* 20 (2011), pp. 569–734 (cit. on p. 149).

[171] B. K. S. Woods, I. Dayyani, and M. I. Friswell. “Fluid/Structure-Interaction Analysis of the Fish-Bone-Active-Camber Morphing Concept”. In: *Journal of Aircraft* 52.1 (Nov. 2014), pp. 307–319 (cit. on p. 132).

[172] B. K. Woods and M. I. Friswell. “Structural Characterization of the Fish Bone Active Camber Morphing Airfoil”. In: *AIAA SciTech Forum*. American Institute of Aeronautics and Astronautics, Jan. 2014 (cit. on p. 132).

[173] B. K. S. Woods and M. I. Friswell. *Preliminary Investigation of a Fishbone Active Camber Concept*. 2012 (cit. on p. 132).

[174] B. K. Woods, O. Bilgen, and M. I. Friswell. “Wind tunnel testing of the fish bone active camber morphing concept”. In: *Journal of Intelligent Material Systems and Structures* 25.7 (Feb. 2014), pp. 772–785 (cit. on p. 132).

[175] B. K. Woods, J. H. Fincham, and M. I. Friswell. “Aerodynamic modelling of the fish bone active camber morphing concept”. In: *Proceedings of the RAeS Applied Aerodynamics Conference, Bristol, UK*. Vol. 2224. 2014 (cit. on p. 132).

[176] P. Woodward and P. Colella. “The numerical simulation of two-dimensional fluid flow with strong shocks”. In: *Journal of Computational Physics* 54 (Apr. 1984), pp. 115–173 (cit. on pp. 74, 75, 77–90, 184).

[177] D. Xu, H. Zhang, Q. Wang, and H. Bao. “Poisson shape interpolation”. In: *Graphical Models* 68.3 (2006). SPM 2005, pp. 268–281 (cit. on pp. 108, 109).

[178] T. Yokozeki, A. Sugiura, and Y. Hirano. “Development of Variable Camber Morphing Airfoil Using Corrugated Structure”. en. In: *Journal of Aircraft* 51.3 (May 2014), pp. 1023–1029 (cit. on p. 116).

[179] M. I. Zafar, F. Fusi, and G. Quaranta. “Multiple input describing function analysis of non-classical aileron buzz”. In: *ADVANCES IN AIRCRAFT AND SPACECRAFT SCIENCE* 4.2 (2017), pp. 203–218 (cit. on p. 157).