A Decade of Progress on Anisotropic Mesh Adaptation for Computational Fluid Dynamics

Frédéric Alauzet, Adrien Loseille

A Decade of Progress on Anisotropic Mesh Adaptation for Computational Fluid Dynamics

Frédéric Alauzet and Adrien Loseille

Gamma3 Team, INRIA Paris-Rocquencourt, 78153 Le Chesnay, France.

Abstract

In the context of scientific computing, the mesh is used as a discrete support for the considered numerical methods. As a consequence, the mesh greatly impacts the efficiency, the stability and the accuracy of numerical methods. The goal of anisotropic mesh adaptation is to generate a mesh which fits the application and the numerical scheme in order to achieve the best possible solution. It is thus an active field of research which is progressing continuously. This review article proposes a synthesis of the research activity of the INRIA Gamma3 team in the field of anisotropic mesh adaptation applied to inviscid flows in computational fluid dynamics since 2000. It shows the evolution of the theoretical and numerical results during this period. Finally, challenges for the next decade are discussed.

Keywords: metric, anisotropic mesh adaptation, continuous mesh framework, computational fluid dynamic, inviscid flows

1. Introduction

1.1. Scientific context

In the field of computational fluid dynamics (CFD), being able to predict numerically all the features of complex flows around or inside complex geometries remains an unachieved goal. However, such an ability would bring opportunities in a better understanding of complex physical phenomena such as turbulence, vorticity, cavitation, etc. From a rational point of view, a lot of engineering, fundamental, and societal problems could then become completely tractable. Indeed, applications range from aerodynamics, aeroacoustics, bio-engineering, and structure design (dam, turbine, oil platform) to citizen safety with blast mitigations. Consequently, numerical simulation, which is now mature, has become an integral part of the design process in science and engineering. The below computational pipeline emphasizes the central role of meshing in scientific computing:

$$\text{CAD} \longrightarrow \text{MESH} \longrightarrow \text{SOLVER} \longrightarrow \text{VISUALIZATION / ANALYSIS} \ .$$

A simple observation of this chart shows: *no mesh = no simulation*. This is why during the late 80's and the 90's a considerable amount of effort was devoted to automatic mesh generation. The state-of-the-art of all the meshing technologies is wide [1], but at present, the main approaches for generality and ease of use in CFD are: multiblock grids [2], adaptive Cartesian grids [3] and unstructured mono-element meshes [4, 5, 6, 7, 8, 9]. This first issue is clear, but a second one is more malicious: *a "bad" mesh implies a wrong or inaccurate solution*. Let us develop this critical issue in scientific computing.

Nowadays, numerical simulations tend to address ever increasing geometrical and physical complexities. The accuracy of geometric models keeps on increasing: if a 10 cm accuracy in the 80's for a business jet was the standard, 1 mm for the geometry and 1 μm in the boundary layer is now commonplace [10, 11]. As shown in Fig. 1, the considered geometries become more and more complex with a lot of tiny details impacting the flows. For instance, for a military aircraft, all major components of the geometry are modeled such as the empennage, rails, missiles,

Email address: {Frederic.Alauzet,Adrien.Loseille}@inria.fr (Frédéric Alauzet and Adrien Loseille)

wings with their leading edge expansions, engine inlet including boundary layer vents, pylons, etc. To illustrate this point, ONERA (the French aerospace lab) flow solver `elsA` historically used block structured hexahedral meshes, and has recently moved to hybrid meshes to be able to handle tiny details in complex geometries [12]. The mesh must indeed be adapted to the geometry.

The other major difficulty arises from the underlying non-linear equations governing the physics, for instance, the Navier-Stokes equations. When the difficulty of the meshing process is overcome, the great complexity of the physics often *requires tailored meshes for which numerical results can be certified* [12, 13]. Such meshes should guarantee the full potential of numerical schemes. However, non-automatic generation of application-fitted meshes leads to time-consuming human intervention. Besides, huge meshes are needed to rely on the numerical solution. In both cases, the CPU time required to run the simulation is prohibitive. Another problem is the certification of industrial simulations. It necessitates at least to perform convergence studies, which require computations on several meshes of different accuracies. Hence, certifying a computation becomes very expensive. Indeed, starting from a 3D mesh of average size h having N vertices, a new mesh with $8N$ vertices is obtained when the mesh size h is divided by a factor 2. The flow solver time step on this new mesh is also divided by 2 as it is proportional to h. Therefore, the CPU time to run the simulation on the new mesh is at least multiplied by 16. If, the mesh size is again divided by two, then the CPU time will be multiplied by a factor 256 as compared to the initial one. If the first simulation has run during 1 day, then it will take almost 9 months to get the result of the last one. High performance computing is a possible answer but, we cannot scale indefinitely. Complementary approaches are needed to reduce the complexity of numerical simulations. **Mesh adaptation** is certainly one of the most powerful tools for this.

Why anisotropic mesh adaptation in CFD ? The observation of flows patterns and the analyze of their characteristics

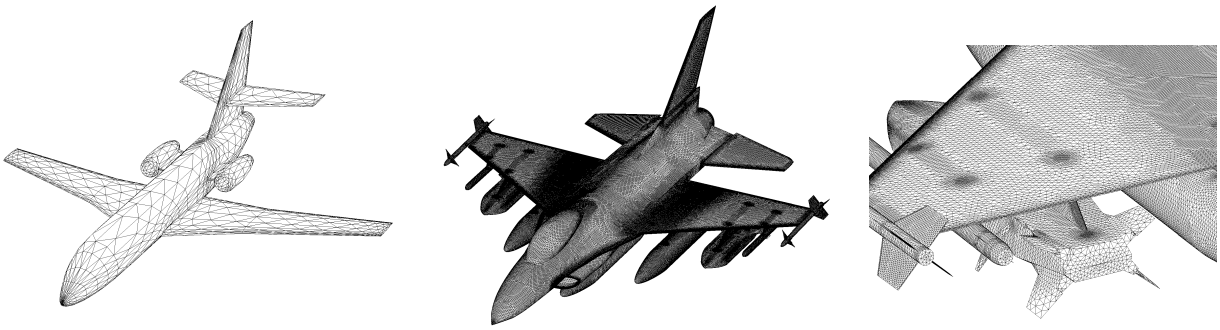


Figure 1: Left, surface mesh of a falcon business jet geometry dating back to the early 90's; it is composed of $1,879$ vertices. Middle, surface mesh of a F-16 fighter geometry mid 2000. It is composed of $271,763$ vertices. Right, close view of the F-16 surface mesh. Note the high accuracy of the details on the geometry.

Figure 2: Left, shock waves (Mach cone) emitted by a supersonic F-14 fighter. Middle, turbulent wake of a transsonic Falcon business jet. Right, schlieren picture showing the flow pattern around a bullet at supersonic speed. Dotted white squares exhibit different components of the flow features: boundary layer near body, shock waves in the far-field and turbulent flow in the wake.

provides a first answer to this question, see Figure 2. In CFD, we notice that physical phenomena of interest (boundary layers, shock waves, contact discontinuities, wake, vortices, etc.) are generally concentrated in small regions of the computational domain. Therefore, uniform meshes are not optimal in terms of *size*. These phenomena are anisotropic by nature, hence uniform meshes are not optimal in terms of *direction*. Thus, it seems natural to take into account the physical phenomena properties inside the mesh in order to improve their representation.

Mathematically speaking, mesh adaptation aims at generating the optimal mesh to control the accuracy of the numerical solution. Optimal signifies that the best possible accuracy is achieved for a given mesh size, or equivalently, a mesh of minimal size is generated to reach a given accuracy. Thus, it enables substantial gains in CPU time, memory requirement, and storage space. The visualization of the results is also facilitated. Furthermore, error estimates have the capability to detect physical phenomena and capture their behavior. Thus, meshes are automatically adapted in critical regions without any *a priori* knowledge of the problem.

In this paper, we will show that mesh adaptation is about more than just improving CPU time. Mesh adaptation enhances flow solver numerical methods because meshes are well suited to the physics. It also enables that ability to certify the quality of the mesh used to compute the desired solution, providing more confidence in the numerical result.

1.2. State-of-the-art

The idea of adapting a mesh is very old. A rather large number of papers have been published dealing with mesh adaptation for numerical simulations. In most of these works, the adaptation is isotropic and done by splitting elements based on predefined patterns. The most powerful idea of anisotropic mesh adaptation emerged later at the end of the 80's thanks to substantial research on error estimates and mesh generation.

In 1987, Peraire et al. proposed a first attempt in 2D by providing error measures involving directions [14]. They pointed out the directional property of the interpolation error and the idea to generate elements with aspect ratios. They considered a local mapping procedure to generate elongated elements. They coupled this with an advancing front technique to generate slightly anisotropic meshes, *i.e.,* a 1:5 ratio.

Similar approaches have been considered by Löhner [15], Selmin and Formaggia [16]. The first attempts in 3D were proposed in the early 90's by Löhner [15] and Peraire et al. [17], but numerical results were almost isotropic and the mesh anisotropy was not clearly visible. In 1994, Zienkiewicz and Wu [18] gave a qualified status on the subject. Despite some great success with this new approach, they emphasized that: *"Unfortunately the amount of elongation which can be used in a typical mesh generation by such mapping is small..."* .

Almost at the same time on the meshing side, Mavriplis proposed to generate stretched elements with a Delaunay approach in 2D in order to obtain high-aspect ratio triangles in boundary layers and wake regions required by aeronautic numerical simulations [19]. In this work, the Delaunay triangulation is performed in a locally stretched space: the idea of the metric has almost emerged. The year after, George, Hecht, and Vallet [20] introduced the use of metrics in a 2D Delaunay mesh generator to handle anisotropic adapted meshes. This idea generalized all the previous works. They exhibited that the absolute value of the Hessian is a metric. The edge lengths and element volumes inside the mesh generator were computed in the Riemannian metric space defined by the given metric field. They proposed to generate a unit uniform mesh in the Riemannian metric space, this mesh being adapted and anisotropic in the physical space.

The fruitful idea of a metric was widely exploited for 2D anisotropic mesh adaptation in the 90's and even more today. For instance, among many others, see the works of [21, 22, 23, 24, 25]. In 1997, Baker gave a state-of-the-art [26] and wrote: *"Mesh generation in three dimensions is difficult enough task in the absence of mesh adaptation and it is only recently that satisfactory three-dimensional mesh generators have become available. [...] . Mesh alteration in three dimensions is therefore a rather perilous procedure that should be under taken with care"*. Indeed, 3D meshing is much more complicated as new pathologies occur. Moreover, the bare existence of such 3D meshes is not guaranteed. Doing 3D anisotropic mesh adaptation is even more complicated.

Meshing blocking points have been partly solved by proposing a local remeshing approach to adapt the mesh. These approaches start from an existing mesh and modify the mesh to adapt. Therefore, there is no more existence problem. At the beginning of the 2000's, the first results with truly 3D anisotropic mesh adaptation for applications were published [27, 28, 29, 30, 31, 32, 33].

In the meantime, new more accurate anisotropic error estimates had been proposed: a posteriori estimates [34, 35, 36], a priori estimates [37, 38, 39, 40], and goal-oriented estimates for functional outputs [41, 42, 43, 44].

Nowadays, this technology can be considered mature, and impressive results can be obtained. For instance in [45], a mean anisotropic ratio of 400 and a mean anisotropic quotient of 50, 000 are reported for adapted meshes containing more than 50 million tetrahedra.

Thanks to its generality, mesh adaptation has been applied to various fields of investigation and also used with a large panel of numerical methods. In all cases, it has brought large improvements in terms of accuracy and CPU performances. Just to give some 3D examples, it has been applied successfully to the sonic boom simulation [45], multi-fluid flows [46, 47], blast problems [48], Stefan problems [30], and metal forming processes [49]. Numerical schemes and mesh adaptation have been coupled with Finite Volume [45], Finite Element [47], Stabilized Finite Element [49], and Discontinuous Finite Element [50] methods, to name a few.

Nowadays, there are a lot of meshing softwares based on the metric concept. Let us cite `Bamg` [51] and `BL2D` [52] in 2D, `Yams` [53] for discrete surface mesh adaptation and `EPIC` [54], `Feflo.a` [55], `Forge3d` [4], `Fun3d` [42], `Gamanic3d` [56], `MAdLib` [57], `MeshAdap` [32], `Mmg3d` [58], `Mom3d` [27], `Tango` [29], and `Libadaptivity` [28] in 3D. It is worth mentioning that all these softwares have arisen from different mesh generation methods. The method in [56, 51] is based on a global constrained Delaunay kernel. In [52], the Delaunay method and the frontal approaches are coupled. [53, 55, 57, 58] are based on local mesh modifications, and [4] is based on the minimal volume principle.

1.3. Identified problematics for numerical simulations at the beginning of 2000

Many groups were able to obtain highly anisotropic adapted meshes of high quality for analytical functions. However, when the solution u becomes a numerical solution u_h provided by a flow solver, additional problems arise. We list below all the problems we identified for CFD at the beginning of 2000.

Loss of anisotropy when turning to numerical solutions. To illustrate this issue, we consider the standard metric-based error estimate (see Relation (8)), which is a control of the interpolation error in L^∞ norm, for the accurate capturing of shock-waves inside a scramjet. This is done by considering a recovered Hessian of one variable of the flow field (Mach, pressure, ...). The final result from [22] is shown in Figure 3. If the final adapted mesh seems perfectly anisotropic (left), a closer view around a shock reveals a complete loss of the anisotropy (right) for this non-regular solution. In fact, the prescribed minimal size is reached in almost every singularity, leading to an isotropic mesh refinement. This problem arises due to several causes:

- an L^∞ norm error estimate has been used. Indeed, let us consider the Heaviside function with a $\delta > 0$ step: $u(x) = \delta$ if $x > 0$ and $u(x) = 0$ if $x \leq 0$, on the segment $[-1, 1]$. Given a uniform mesh of this domain with size parameter h, then we easily demonstrate that the interpolation error in L^p norm converges at order $O(h^{\frac{1}{p}})$. Therefore, the spatial convergence order is $O(h^{\frac{1}{p}})$. When p tends to infinity, the expected convergence order is $O(1)$. In other words, the L^∞ norm will never converge in presence of discontinuities. The algorithm does not diverge due to the prescription of a minimal size. Thus, an L^∞ error estimate is not suitable for flows with discontinuities. Consequently, L^p strategies become of main interest in the case of discontinuous solutions

- we have no guarantee that the recovery techniques used to compute the Hessians provide a smooth metric field, even more for a non-regular flow field. Then, it becomes difficult for the adaptive mesher to work in this irregular metric space which diminishes the resulting anisotropy of the mesh

- a non-oscillatory flow solver is required on anisotropic adapted meshes. If the numerical methods present oscillations in shocks (no TVD property), then variations are captured in the direction parallel to the shock wave, which leads to a drastic reduction of the mesh anisotropy. The solver stabilization must therefore be efficient on anisotropic meshes.

Capturing all scales of the flow field. A second problem is that a control of the interpolation error in L^∞ does not capture the small-scale features of the flow. Several modifications of the L^∞ interpolation error estimate have been considered [22, 59, 60] to overcome this issue (see Relation (9)). However, such local normalizations are only slightly more sensitive and the control of the interpolation error remains in L^∞ norm.

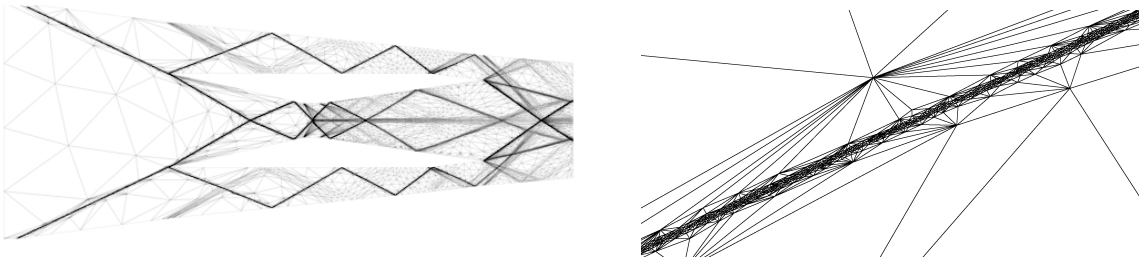


Figure 3: Scramjet adaptive computation based on a control of the interpolation error in L^∞ norm [22].

Solution accuracy spoiled by interpolations. Solution interpolation is also a key stage in the context of mesh adaptation. Indeed, it links the mesh generation and the numerical flow solver allowing the simulation to be restarted from the previous state. This recurrent stage in adaptive simulations is crucial for time-dependent problems (at least for problems with advection) as the errors introduced by this procedure accumulate throughout the computations. The impact of such errors on the solution accuracy was pointed out with standard linear interpolation. Consequently, interpolation schemes taking into account the properties of the modeled flows need to be found.

Adaptive meshers robustness. When a global strategy is employed, the problem of existence of the mesh has to be solved at each remeshing phase. If this procedure fails, then the whole adaptive loop stops, leaving the user without any solution. Knowing that this issue is even more complex when adapted anisotropic surface meshes are considered, different remeshing strategies have to be considered.

Some conclusions. These issues illustrate that each step of the mesh adaptation loop must be up-to-date to avoid failure and to achieve the best efficiency. None of these stages can be neglected. If one stage of the adaptation loop fails then all adaptive processes collapse.

1.4. Outline

This review paper is organized as follows. We first succinctly recall the metric-based mesh generation in Section 2, then Section 3 gives the theoretical framework of anisotropic mesh adaptation which is based on the continuous mesh concept. Section 4 describes the application of anisotropic mesh adaptation to steady inviscid flow computations in Computational Fluid Dynamics (CFD), and Section 5, its extension to time-dependent problems. We conclude this article with some remaining challenges for the next decade in Section 6.

2. Metric-based mesh generation

The basics of metric-based mesh adaptation such as metric notion, Riemannian metric space, computation of geometric quantities in metric space, and operations on metrics can be found in [61, 62, 63]. Here, we just give an overview of the metric-based mesh generation.

To generate anisotropic meshes, one must be able to prescribe at each point of the domain privileged sizes and orientations for the elements. The use of Riemannian metric spaces is an elegant and efficient way to achieve this goal. The main idea of metric-based mesh adaptation, initially introduced in [20], is to generate a unit mesh in a prescribed Riemannian metric space.

A tetrahedron K, defined by its list of edges $(\mathbf{e_i})_{i=1..6}$, is **unit** with respect to a metric tensor $\mathcal{M}$ if the length of all its edges is unit in metric $\mathcal{M}$:

$$\forall i = 1, ..., 6, \quad \ell_{\mathcal{M}}(\mathbf{e_i}) = 1 \text{ with } \ell_{\mathcal{M}}(\mathbf{e_i}) = \sqrt{{}^t\mathbf{e_i}\,\mathcal{M}\,\mathbf{e_i}}. \tag{1}$$

If all edges of K are of unit length, then its volume $|K|_{\mathcal{M}}$ in $\mathcal{M}$ is constant equal to:

$$|K|_{\mathcal{M}} = \frac{\sqrt{2}}{12} \quad \text{and} \quad |K| = \frac{\sqrt{2}}{12}(\det(\mathcal{M}))^{-\frac{1}{2}}, \tag{2}$$

where $|K|$ is its Euclidean volume.

The notion of **unit mesh** is far more complicated than the notion of unit element as the existence of a mesh composed only of unit regular simplices with respect to a given Riemannian metric space is not guaranteed. Therefore, the notion of unit mesh has to be relaxed. A discrete mesh $\mathcal{H}$ of a domain $\Omega \subset \mathbb{R}^n$ is a **unit mesh** with respect to Riemannian metric space $\mathbf{M} = (\mathcal{M}(\mathbf{x}))_{\mathbf{x}\in\Omega}$ if all its elements are quasi-unit. The definition of unity for an element is relaxed by taking into account technical constraints imposed by mesh generators. For instance, to avoid cycling while analyzing edges lengths in a local remesher, a tetrahedron K defined by its list of edges $(\mathbf{e_i})_{i=1...6}$ is said to be quasi-unit if, $\forall i,\ \ell_{\mathcal{M}}(\mathbf{e_i}) \in [\frac{1}{\sqrt{2}}, \sqrt{2}]$ and it has a unit volume.

Using the previous notions, the problem of mesh generation can be considerably simplified. Indeed, whatever the kind of desired mesh (uniform, adapted isotropic, adapted anisotropic), the mesh generator will always generate a unit mesh in the prescribed Riemannian metric space [20]. Consequently, the generated mesh is uniform and isotropic in the Riemannian metric space while it is adapted and anisotropic in the Euclidean space. This is illustrated in Figure 4. This idea has turned out to be a huge breakthrough in the generation of anisotropic adapted meshes.

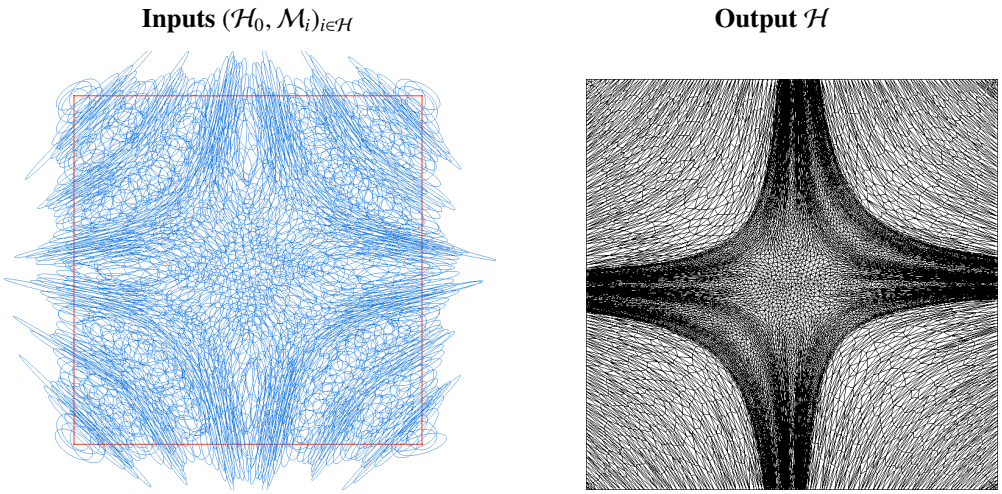


Figure 4: Metric-based mesh generation. Left, specified Riemannian metric space. Right, unit mesh in the prescribed Riemannian metric space which becomes adapted anisotropic in the Euclidean space.

3. Continuous mesh framework

The previous section has emphasized the role of metric tensors and Riemannian metric spaces as useful mathematical tools to prescribe sizes and directions to adaptive meshers. Additionally, these differential geometry notions are more than just a simple tool for mesh generation. In this section, we go further in this analysis and we demonstrate that there is a comprehensive duality between meshes and Riemannian metric spaces. More precisely, Riemannian metric spaces can be seen as continuous models representing meshes. The fundamental consequence is that all kind of mathematical analysis can be performed using such spaces for which powerful mathematical tools are available. Even mathematical problems which could not even be considered on discrete meshes can be addressed in this new framework. More details can be found in [64, 62, 65, 39].

3.1. Motivation

As the use of Riemannian metric fields is well-suited for the generation of anisotropic adapted meshes, we must be able to design suitable metric tensor fields from error estimates. When the solution is assumed to be smooth, a straightforward solution consists in controlling the linear interpolation error. As the linear interpolation error involves the Hessian of the solution, a metric-based estimate is easily derived, see [14, 21, 22, 38, 60]. When solutions are of low regularity, *a priori* or *a posteriori* error estimates need to be modified to come up with a metric tensor

field, as in [37, 34]. However, both approaches generally involve successive bounds to end up with a metric-based estimate. There are neither specific study on the importance of these approximations nor any guarantee of the veracity of the provided upper bound. In this context, we can still wonder if using metric tensor fields to control such error estimates is a relevant choice. For the linear interpolation error, a lot of numerical examples for real life problems [29, 24, 35, 28, 66, 27, 60, 65] tend to answer affirmatively to this question. In this section, this question is theoretically addressed and the pertinence of the use of metric tensor fields for anisotropic mesh adaptation is assessed.

3.2. Duality between discrete and continuous entities

The following points out the strong duality between discrete entities, e.g. elements and meshes, and continuous mathematical objects, e.g. metric tensors and Riemannian metric spaces.

Proposition 3.1. *Let $\mathcal{M}$ be a metric tensor, there exists a non-empty infinite set of unit elements with respect to $\mathcal{M}$. Conversely, given an element K such that $|K| \neq 0$, there is a unique metric tensor $\mathcal{M}$ for which element K is unit with respect to $\mathcal{M}$.*

The consequence of this proposition is that the function *unit with respect to* defines classes of equivalences of discrete elements. Thus, in the continuous mesh framework, a metric tensor $\mathcal{M}$ is called **continuous element**. It is used to model all discrete elements that are unit for $\mathcal{M}$. Geometric quantities associated with a continuous element can be computed. For instance, according to Relation (2), the volume of a continuous element is given by $\frac{\sqrt{2}}{12}(\det(\mathcal{M}))^{-\frac{1}{2}}$. Other geometric properties are given in [62].

Similarly, in the continuous mesh framework, a **continuous mesh** of a domain Ω is defined by a collection of continuous elements $\mathbf{M} = (\mathcal{M}(\mathbf{x}))_{\mathbf{x}\in\Omega}$, *i.e.,* a Riemannian metric space. It is used to model all meshes that are unit for $\mathbf{M}$. The properties of the continuous mesh can be exhibited by rewriting $\mathbf{M}$ in order to distinguish local properties from global ones:

Proposition 3.2. *A Riemannian metric space $\mathbf{M} = (\mathcal{M}(\mathbf{x}))_{\mathbf{x}\in\Omega}$ locally writes:*

$$\forall \mathbf{x} \in \Omega, \quad \mathcal{M}(\mathbf{x}) = d^{\frac{2}{3}}(\mathbf{x})\, \mathcal{R}(\mathbf{x}) \begin{pmatrix} r_1^{-\frac{2}{3}}(\mathbf{x}) & & \\ & r_2^{-\frac{2}{3}}(\mathbf{x}) & \\ & & r_3^{-\frac{2}{3}}(\mathbf{x}) \end{pmatrix} {}^t\mathcal{R}(\mathbf{x}),$$

where

- *density d is equal to: $d = (\lambda_1\lambda_2\lambda_3)^{\frac{1}{2}} = (h_1h_2h_3)^{-1}$, with λ_i the eigenvalues of $\mathcal{M}$*
- *anisotropic quotients r_i are equal to: $r_i = h_i^3\,(h_1h_2h_3)^{-1}$*
- *$\mathcal{R}$ is the eigenvectors matrix of $\mathcal{M}$ representing the orientation.*

The density d controls only the local level of accuracy of $\mathbf{M}$. Increasing or decreasing d does not change the anisotropic properties nor the orientation. The anisotropy property is given by the anisotropic quotients r_i and the orientation by matrix $\mathcal{R}$. We also define the complexity C of $\mathbf{M}$:

$$C(\mathbf{M}) = \int_\Omega d(\mathbf{x})\, \mathrm{d}\mathbf{x} = \int_\Omega \sqrt{\det(\mathcal{M}(\mathbf{x}))}\, \mathrm{d}\mathbf{x}.$$

This real-value parameter is useful to quantify the global level of accuracy of $(\mathcal{M}(\mathbf{x}))_{\mathbf{x}\in\Omega}$. It can also be interpreted as the continuous counterpart of the number of vertices of a discrete mesh.

3.3. Continuous linear interpolation error: discrete-continuous duality

Now, we aim at applying this framework in the context of error estimation. However, as our intent is to propose a fully discrete-continuous duality, it is not enough to derive only the optimal mesh arising from an interpolation error bound as in classical studies on interpolation error. Instead, we want to evaluate the interpolation error for any function on any continuous mesh without imposing any optimality condition as alignment or equi-distribution, ...

Let $\mathbf{M} = (\mathcal{M}(\mathbf{x}))_{\mathbf{x}\in\Omega}$ be a continuous mesh of a domain Ω and let u be a non-linear function which is assumed to be twice continuously differentiable. We seek a well-posed definition of the continuous linear interpolation error $\|u - \pi_{\mathcal{M}}u\|_{L^1(\Omega)}$ related to a continuous mesh $(\mathcal{M}(\mathbf{x}))_{\mathbf{x}\in\Omega}$ which implies a well-posed definition of a linear continuous interpolate $\pi_{\mathcal{M}}u$. More precisely, we would like the continuous linear interpolation error to be a reliable mathematical model of $\|u - \Pi_h u\|_{L^1(\Omega_h)}$ where Π_h is defined by a mesh $\mathcal{H}$ of a discretized domain Ω_h which is a unit mesh with respect to $(\mathcal{M}(\mathbf{x}))_{\mathbf{x}\in\Omega}$. This means that considering $\|u - \pi_{\mathcal{M}}u\|_{L^1(\Omega)}$ is equivalent to consider $\|u - \Pi_h u\|_{L^1(\Omega_h)}$.

Local interpolation error duality. We consider a quadratic function u defined on a domain $\Omega \subset \mathbb{R}^3$. The function is given by its matrix representation:

$$\forall \mathbf{x} \in \Omega, \quad u(\mathbf{x}) = \frac{1}{2}\, {}^t\mathbf{x}\, H\, \mathbf{x},$$

where H is a symmetric matrix representing the Hessian of u. For any symmetric matrix H, $|H|$ denotes the positive symmetric matrix deduced from H by taking the absolute values of its eigenvalues. The function u is linearly interpolated on a tetrahedron K. We denote by $\Pi_h u$ the linear interpolate of u on K. If K is now assumed to be unit with respect to $\mathcal{M}$, we can state the following result:

Theorem 3.1. *For all unit elements K with respect to $\mathcal{M}$, the interpolation error of u in L^1 norm does not depend on the element shape and is only a function of the Hessian H of u and of $\mathcal{M}$. In 3D, for all unit tetrahedra K with respect to $\mathcal{M}$, the following equality holds:*

$$\|u - \Pi_h u\|_{L^1(K)} = \frac{\sqrt{2}}{240}\, \det(\mathcal{M}^{-\frac{1}{2}})\, \mathrm{trace}(\mathcal{M}^{-\frac{1}{2}}\, H\, \mathcal{M}^{-\frac{1}{2}}). \tag{3}$$

Relation (3) shows that the infinite set of discrete elements that are unit for a given continuous element $\mathcal{M}$ achieves the same interpolation error. Moreover, it shows that this interpolation error is only expressed with continuous quantities: the continuous element $\mathcal{M}$ and the Hessian of the function u. Consequently, Theorem 3.1 points out that the metric alone contains enough information to describe completely the linear interpolation error in L^1 norm. In other words, this theorem confirms that the use of metric-based mesh adaptation is particularly well suited to control anisotropically the interpolation error.

Continuous linear interpolate. The main difficulty in defining the continuous linear interpolate is to connect a discrete error computed on an element to a local continuous error model that is defined point-wise. Indeed, the discrete interpolation error in norm L^1 is integrated on the element K, whereas a continuous mesh is a function $\mathbf{x} \mapsto \mathcal{M}(\mathbf{x})$ defined at each point $\mathbf{x}$ of Ω.

Suppose now that the continuous mesh $(\mathcal{M}(\mathbf{x}))_{\mathbf{x}\in\Omega}$ is varying and that the function u is no more quadratic but only twice continuously differentiable. If Equality (3) of Theorem 3.1 does not hold anymore, all the terms of the right-hand-side $\mathcal{M}$ and H are well defined continuously. The definition of a continuous interpolate follows up from this consideration.

In the vicinity of $\mathbf{a}$, u_Q is the quadratic approximation of smooth function u and $(\mathcal{M}(\mathbf{x}))_{\mathbf{x}\in\Omega}$ reduces to $\mathcal{M}(\mathbf{a})$ in the tangent space. We can now state the main result providing an equivalence formula between discrete and continuous views locally around a point $\mathbf{a}$ of the domain:

Theorem 3.2. *Let u be a twice continuously differentiable fonction of a domain Ω and $(\mathcal{M}(\mathbf{x}))_{\mathbf{x}\in\Omega}$ be a continuous mesh of Ω. Then, there exists a unique function $\pi_{\mathcal{M}}$ such that:*

$$\forall \mathbf{a} \in \Omega\,, \quad |u - \pi_{\mathcal{M}}u|(\mathbf{a}) \;=\; \frac{\|u_Q - \Pi_h u_Q\|_{L^1(K)}}{|K|} \;=\; \frac{1}{20}\, \mathrm{trace}(\mathcal{M}(\mathbf{a})^{-\frac{1}{2}}\, |H(\mathbf{a})|\, \mathcal{M}(\mathbf{a})^{-\frac{1}{2}})\,,$$

for every K unit element with respect to $\mathcal{M}(\mathbf{a})$ and where u_Q is the quadratic model of u at $\mathbf{a}$.

This theorem underlines another discrete-continuous duality by pointing out a continuous counterpart of the interpolation error. For this reason, we propose the following formalism: $\pi_{\mathcal{M}}$ is called *continuous linear interpolate* and $|u - \pi_{\mathcal{M}} u|$ represents the continuous dual of the interpolation error.

The local interpolation error duality becomes global when the mesh is unit with respect to a constant metric tensor (this does not necessary imply that the mesh is uniform) and when the function is quadratic. In this specific case, by neglecting errors due to the boundary discretization, we have the equality:

$$\|u - \Pi_h u\|_{L^1(\Omega_h)} = \|u - \pi_{\mathcal{M}} u\|_{L^1(\Omega)},$$

for all meshes $\mathcal{H}$ which are unit with respect to $(\mathcal{M}(\mathbf{x}))_{\mathbf{x}\in\Omega}$.

Theoretically, the use of a discrete support is no more mandatory to compute the interpolation error. The example below shows that the computation of the discrete interpolation error on a unit mesh $\mathcal{H}$ with respect to $\mathbf{M}$ is an accurate approximation of the continuous interpolation error calculated on $\mathbf{M}$, even for non quadratic functions and non-constant continuous meshes.

Continuous example. The set of 2D continuous meshes $\mathbf{M}(\alpha) = (\mathcal{M}_\alpha(\mathbf{x}))_{\mathbf{x}\in\Omega}$ is defined on square domain $\Omega = [0,1]\times[0,1]$ by:

$$\mathcal{M}_\alpha(x,y) = \alpha \begin{pmatrix} h_1^{-2}(x,y) & 0 \\ 0 & h_2^{-2}(x,y) \end{pmatrix},$$

where $h_1(x,y) = 0.1(x+1) + 0.05(x-1)$ and $h_2(x,y) = 0.2$. Parameter α is used to control the level of accuracy of the mesh. The continuous mesh becomes coarser when α decreases but anisotropic quotients and orientations remain constant. This trend is given by the computation of complexity $C(\mathbf{M}(\alpha))$:

$$C(\mathbf{M}(\alpha)) = N(\alpha) = \iint_\Omega \frac{1}{h_1 h_2}(x,y)\,\mathrm{d}x\mathrm{d}y = \frac{200}{3}\ln(2)\alpha.$$

The continuous interpolation error on $\mathbf{M}(\alpha)$ is computed for two analytical functions: $u_1(x,y) = 6x^2 + 2xy + 4y^2$ and $u_2(x,y) = \mathrm{e}^{(2x^2+y)}$. As regards function u_1, the point-wise continuous interpolation error on $\mathbf{M}(\alpha)$ is

$$(u_1 - \pi_{\mathcal{M}} u_1)(x,y) \;=\; \frac{27\,x^2 + 18\,x + 35}{800\,\alpha}.$$

The previous expression is then integrated over Ω:

$$\iint_\Omega |u_1 - \pi_{\mathcal{M}} u_1|(x,y)\,\mathrm{d}x\mathrm{d}y \;=\; \frac{53}{800\,\alpha} \;=\; \frac{53}{12}\frac{\ln(2)}{N(\alpha)}.$$

For function u_2, the point-wise continuous interpolation error on $\mathbf{M}(\alpha)$ is:

$$(u_2 - \pi_{\mathcal{M}} u_2)(x,y) \;=\; \frac{e^{4x^2+y}}{8\,\alpha}\left((0.15x + 0.05)^2\,(4 + 16x^2) + 0.05\right).$$

By direct integration over Ω, it comes:

$$\iint_\Omega |u_2 - \pi_{\mathcal{M}} u_2|(x,y)\,\mathrm{d}x\mathrm{d}y \;\approx\; \frac{0.2050950191}{\alpha} \;\approx\; \frac{13.673\,\ln(2)}{N(\alpha)}.$$

These analytical evaluations of the continuous interpolation error are compared to the evaluation of the discrete interpolation error on a set of a unit meshes with respect to $\mathbf{M}(\alpha)$ for several values of α. These evaluations are plotted in Figure 5 where a perfect correlation is observed. The black-plain lines represent the extremal bound of the interpolation error due to the relaxed notion of quasi-unit elements while considering uniform meshes in $\mathbf{M}(\alpha)$ with edges length equal to $\sqrt{2}/2$ (lower line) and 2 (upper line).

A study with several other continuous examples is presented in [64].

3.4. Optimal control of the interpolation error in L^p norm

In its more general form, the problem of mesh adaptation consists in finding the mesh $\mathcal{H}$ of a domain Ω that minimizes a given error for a given function u. For the sake of simplicity, we consider here the linear interpolation error $u - \Pi_h u$ controlled in L^p norm. Note that considering other norms also works [38]. The problem is thus stated in an *a priori* way:

$$\text{Find } \mathcal{H}_{opt} \text{ having } N \text{ vertices such that } \mathbf{E}_{L^p}(\mathcal{H}_{opt}) = \min_{\mathcal{H}} \|u - \Pi_h u\|_{L^p(\Omega_h)}. \tag{P}$$

(P) is a global combinatorial problem which turns out to be intractable practically. Indeed, this would require the simultaneous optimization of both the mesh topology and the vertices location. Consequently, simpler problems are considered to approximate the solution. A common simplification is to perform a local analysis of the error instead of considering the global problem. A first set of methods consists in deriving the optimal element shape [67, 68]. A second set consists in deriving a local bound of the interpolation error. This bound is then transformed into a metric-based estimate [37, 38, 34, 60]. Direct minimization of the error can also be considered by using directly the interpolation error as a cost function in the mesh generator [69]. All these strategies have in common the resolution of a local problem as they act in the vicinity of an element. Consequently, such error minimizations are equivalent to a steepest descent algorithm that converges only to a local minimum with poor convergence properties. This drawback arises because a minimization on a discrete mesh is directly considered.

We propose to address the resolution of (P) in a continuous setting. Consequently, (P) is recast as a continuous optimization problem where the discrete interpolation error is replaced by the continuous one:

$$\text{Find } \mathbf{M}_{opt} \text{ having a complexity of } N \text{ such that } \mathbf{E}_{L^p}(\mathbf{M}_{opt}) = \min_{\mathbf{M}} \|u - \pi_{\mathcal{M}} u\|_{L^p(\Omega)}.$$

Contrary to discrete-based studies, the continuous formulation succeeds in solving globally the optimal interpolation error problem by using powerful mathematical tools such as calculus of variations.

Optimal continuous mesh. Using the definition of the linear continuous interpolate $\pi_{\mathcal{M}}$, it is then possible to set the well-posed global optimization problem of finding the optimal continuous mesh minimizing the continuous interpolation error in L^p norm:

$$\begin{aligned} \text{Find } \mathbf{M}_{L^p} = \min_{\mathbf{M}} \mathbf{E}_{L^p}(\mathbf{M}) &= \left(\int_\Omega (u(\mathbf{x}) - \pi_{\mathcal{M}} u(\mathbf{x}))^p \, \mathrm{d}\mathbf{x} \right)^{\frac{1}{p}} \\ &= \left(\int_\Omega \mathrm{trace}\Big(\mathcal{M}(\mathbf{x})^{-\frac{1}{2}} |H_u(\mathbf{x})| \mathcal{M}(\mathbf{x})^{-\frac{1}{2}} \Big)^p \, \mathrm{d}\mathbf{x} \right)^{\frac{1}{p}}, \end{aligned} \tag{4}$$

under the constraint $C(\mathbf{M}) = \displaystyle\int_\Omega d(\mathbf{x}) \, \mathrm{d}\mathbf{x} = N$.

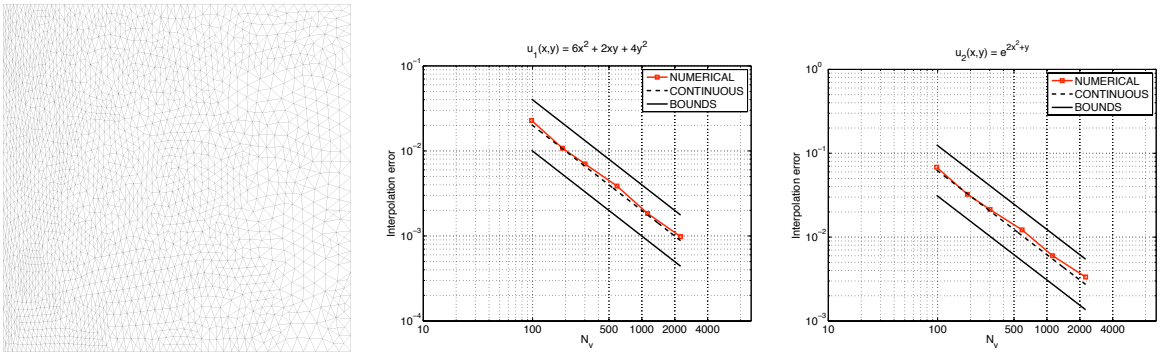


Figure 5: Left, a unit mesh with respect to $\mathbf{M}(\alpha)$ for $\alpha = 32$. Comparison between continuous interpolation error $\|u - \pi_{\mathcal{M}} u\|_{L^1(\Omega)}$ and discrete interpolation error $\|u - \Pi_h u\|_{L^1(\Omega)}$ evaluations for functions u_1 (middle) and u_2 (right) on $\mathbf{M}(\alpha)$.

The constraint on the complexity is added to avoid the trivial solution where all $(h_i)_{i=1,3}$ are zero which provides a null error. Contrary to a discrete analysis, this problem can be solved globally by using calculus of variations that is well-defined on the space of continuous meshes. In [64], it is proved that Problem (4) admits a unique solution. In addition, the following properties hold:

Theorem 3.3. *Let u be a twice continuously differentiable function defined on $\Omega \subset \mathbb{R}^3$, H_u its Hessian, the optimal continuous mesh $\mathbf{M}_{L^p}(u) = (\mathcal{M}_{L^p}(x))_{x\in\Omega}$ minimizing Problem (4) locally reads:*

$$\mathcal{M}_{L^p}(\mathbf{x}) = N^{\frac{2}{3}} \left(\int_\Omega \det(|H_u(\bar{\mathbf{x}})|)^{\frac{p}{2p+3}} \, \mathrm{d}\bar{\mathbf{x}} \right)^{-\frac{2}{3}} \det(|H_u(\mathbf{x})|)^{\frac{-1}{2p+3}} \, |H_u(\mathbf{x})| \, . \tag{5}$$

It verifies the following properties:

- $\mathbf{M}_{L^p}(u)$ *is unique*
- $\mathbf{M}_{L^p}(u)$ *is locally aligned with the eigenvectors basis of H_u and has the same anisotropic quotients as H_u*
- $\mathbf{M}_{L^p}(u)$ *provides an optimal explicit bound of the interpolation error in L^p norm:*

$$\|u - \pi_{\mathcal{M}_{L^p}} u\|_{L^p(\Omega)} = 3\, N^{-\frac{2}{3}} \left(\int_\Omega \det\left(|H_u|\right)^{\frac{p}{2p+3}} \right)^{\frac{2p+3}{3p}} . \tag{6}$$

- *For a sequence of continuous meshes having an increasing complexity with the same orientation and anisotropic quotients $(\mathbf{M}^N_{L^p}(u))_{N=1\ldots\infty}$, the asymptotic order of convergence verifies:*

$$\|u - \pi_{\mathcal{M}^N_{L^p}} u\|_{L^p(\Omega)} \leq \frac{Cst}{N^{2/3}}. \tag{7}$$

  Relation (7) points out a global second order of mesh convergence.

Note that passing to the limit for $p \to \infty$ leads to the classical metric given by Relation (8) that controls the interpolation error in L^∞ norm as used in [22, 60].

Analytical example. For analytic functions, the mesh adaptation algorithm is the following:

Algorithm 1 Mesh Adaptation Loop for Analytical Function

Initial mesh $\mathcal{H}_0$ and targeted complexity N
For $i = 1, n_{adap}$

1. $\{f_{i-1}, H_{f_{i-1}}\}$ = Evaluate f and its Hessian H_f on mesh $\mathcal{H}^{i-1}$;
2. $(\mathcal{M}_{L^p,i-1})$ = Compute metric $\mathcal{M}_{L^p}$ according to Relation (5);
3. $\mathcal{H}_i$ = Generate a new adapted mesh from pair $(\mathcal{H}_{i-1}, \mathcal{M}_{L^p,i-1})$;

EndFor

This example aims at exemplifying the second order mesh convergence as well as illustrating the different behaviors of the L^p norms by quantifying the impact of the local normalization term $(\det|H_u|)^{\frac{2p+3}{p}}$. We consider the smooth function f defined on $[-1,1]\times[-1,1]$:

$$f(x,y) = \begin{cases} 0.01\sin(xy) & \text{if } xy < \frac{\pi}{50} \\ \sin(50xy) & \text{if } \frac{\pi}{50} \leq xy \leq 2\frac{\pi}{50} \\ 0.01\sin(50xy) & \text{if } xy > 2\frac{\pi}{50} \end{cases}$$

f is composed of small and large scales with an amplitude of 0.01 and 1, Figure 6. Mesh adaptations based on L^1, L^2 and L^4 norms are performed. As predicted by the theory, the second order mesh convergence is reached for all norms. More than $10,000$ vertices are required to reach this asymptotic order of convergence. The convergence plot, Figure 6, can be decomposed into three parts:

(i) a second order mesh convergence part (from 100 to 2,000 vertices). In this region, only the main sinusoidal variations are accurately captured whereas small scales are ignored. Consequently, a second order mesh convergence arises from these main scales

(ii) a first order mesh convergence part (from 2,000 to 10,000 vertices) where the mesh size is small enough to capture small variations of the function. At this time, the error committed on these small fluctuations becomes preponderant

(iii) the final asymptotic second order convergence part (from 10,000 to 100,000 vertices) when all scales of the function are captured by the mesh.

As p increases, the second part is shifted to the right, or equivalently, the number of nodes increases. Consequently, the L^1 norm captures these small scales earlier. For instance, if we look at adapted meshes containing $7,000$ vertices, small scales are well refined with L^1 and L^2 norms whereas they are ignored by the L^4 norm, see Figure 7.

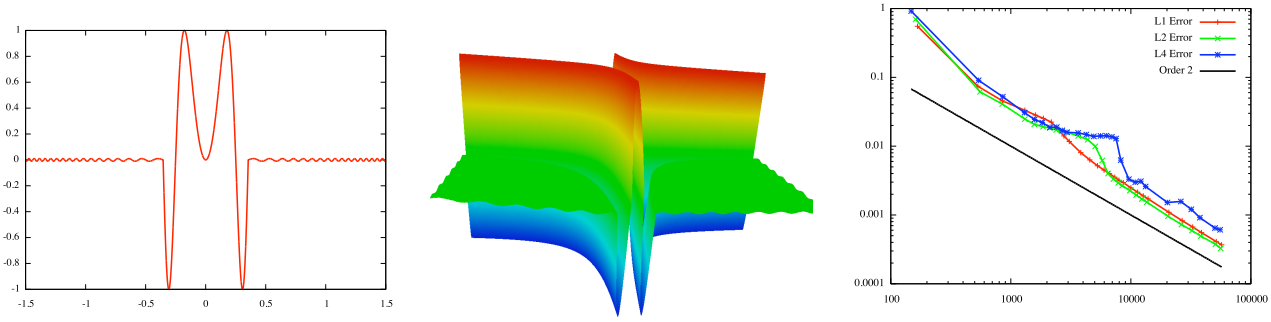


Figure 6: Smooth function f with tiny and large variation. Left, cut through the straight line $y = x$ and, middle, function iso-values and amplitudes. Right, mesh convergence estimates for the function f with mesh adaptations in L^1, L^2 and L^4 norms.

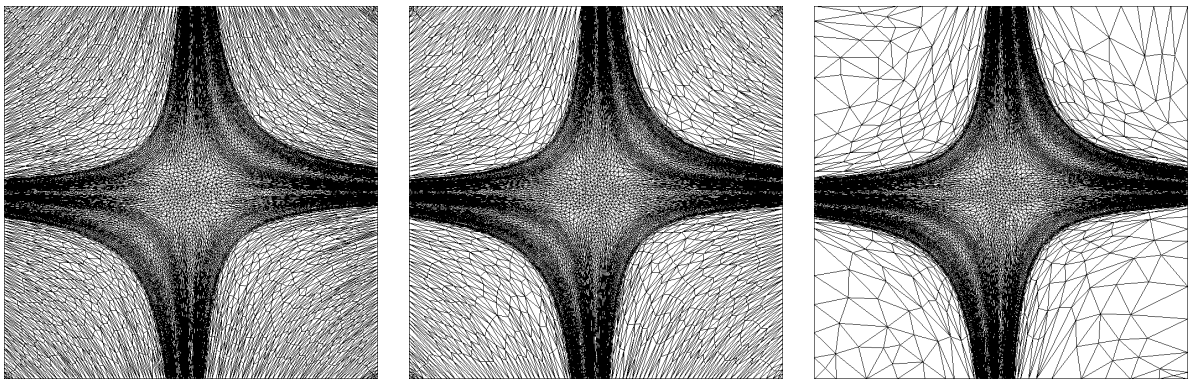


Figure 7: Adapted meshes to f for L^p norms with $p = [1, 2, 4]$ from left to right. Each mesh contains almost $7,000$ vertices. In this range of vertices, small scales are only captured by the L^1 and L^2 norms.

3.5. Chronology of the results

We can quickly recall the evolution of the interpolation error estimate. Initially, the optimal metric was defined to equally distribute the interpolation error over the mesh [20] and given by:

$$\mathcal{M}_{L^\infty} = \frac{c_d}{\varepsilon}|H_u|\,. \tag{8}$$

where c_d is a constant depending on the dimension and ε is the error threshold. The problem is that such estimate was not able to accurately capture all the scales of the solution as large amplitude phenomena (like shock waves

and boundary layer) tend to mask weaker phenomena (like vortices and recirculation). Several modifications of this Hessian-based estimate have been considered to overcome this issue. For instance, the following local normalizations:

$$\mathcal{M}_{L^\infty} = \frac{|H|}{|u|}, \quad \mathcal{M}_{L^\infty} = \frac{|H|}{|u| + h\,||\nabla u||_2}, \quad \mathcal{M}_{L^\infty} = \frac{|H|}{\gamma|u| + (1-\gamma)h||\nabla u||_2}, \tag{9}$$

were introduced in [22, 59, 60], respectively, as an attempt to capture all the scales of the solution. However, such local normalizations are only slightly more sensitive and the control of the interpolation error remains in L^∞ norm.

Moreover, a control of the interpolation error in L^∞ norm does not converge in discontinuities and thus leads to a loss of anisotropy in such regions [65]. All the benefits of anisotropic mesh adaptation are lost.

A control of the interpolation error in L^p norm solves all these issues and is of main interest in the case of multiscale or discontinuous solutions. The optimal metric is given by Relation (5):

$$\mathcal{M}_{L^p} = D_{L^p}\ \det(|H_u|)^{\frac{-1}{2p+3}}\ |H_u|\,.$$

4. Metric-based anisotropic mesh adaptation for steady flows

This section presents two metric-based anisotropic mesh adaptation approaches for steady problems in CFD :

- The first method is the **feature-based** anisotropic mesh adaptation (Section 4.3). The goal is *to derive the best mesh to compute the characteristics of a given sensor* w. Many works have been done on this topic, among them [27, 28, 34, 35, 29, 32, 60, 31, 46]...
- The second method is the **goal-oriented** anisotropic mesh adaptation (Section 4.4). It aims at *deriving the best mesh to observe a given scalar functional* $j(w) = (g, w)$. Less work are available in the literature as the anisotropy is hard to prescribe in such framework [41, 42, 70, 71, 72, 73].

Before introducing these two kinds of error estimate, we give the problems encountered at the end of the 90's, and we present the mesh adaptation loop for numerical simulations.

4.1. Problems and motivations

At the end of the 90's, a lot of successful examples on real-life problems have proved the efficiency of metric-based anisotropic mesh adaptation to improve the ratio between solution accuracy and the number of degrees of freedom. Nevertheless, several issues remain in order to attain efficient adaptive methods for numerical computations and to assess adaptive computations. They are mainly related to the fact that we are now working with numerical solutions. For instance, several questions remain on the optimality of the mesh, the assessment of the numerical solution, the convergence of the computation according to the expected theoretical order, the automatic detection and capturing of all the scales of the solution, ... Consequently, answering positively to these questions is not straightforward as we face both theoretical and practical difficulties. We briefly recall all the difficulties and the way to cope with them.

Approximation error. In the context of numerical simulations, we seek for the best mesh approximating the smooth solution u of a steady scalar PDE over a domain Ω of $\mathbb{R}^d$. The considered problem of mesh adaptation consists in finding the mesh $\mathcal{H}$ of Ω that minimizes the approximation error $u - u_h$ in L^p norm. The problem is thus stated in an *a priori* way:

$$\text{Find } \mathcal{H}_{L^p} \text{ having } N \text{ vertices such that } E_{L^p}(\mathcal{H}_{L^p}) = \min_{\mathcal{H}} \left(\int_{\Omega_h} (u(\mathbf{x}) - u_h(\mathbf{x}))^p \, \mathrm{d}\mathbf{x} \right)^{\frac{1}{p}}. \tag{10}$$

As compared to the previous section, the solution u is unknown and we need to control the error between the solution and its numerical approximation u_h provided by a numerical flow solver. We have to specify the link between the approximation error and the interpolation error.

Efficient anisotropic mesh adaptation for numerical simulations. We have seen in Section 1.3 two main problems which considerably limit the efficiency of anisotropic mesh adaptation: (i) loss of anisotropy and (ii) capture of all scales of the flow.

As already observed in the previous section, considering a control of the interpolation error in L^p norm enables the ability to *capture all the scales* of analytical functions. The same behavior for numerical solutions involving multiscale physical phenomena is therefore expected. Indeed, numerical experiments show that solution scales that have an amplitude $1,000$ times lower than the largest one are still captured and refined. The use of L^p norm instead of L^∞ one also enables the ability to *achieve convergence in singularities*. From a practical point of view, prescribing a *minimal size is no longer required*. This results in the generation of highly anisotropic meshes.

However, canceling out loss of anisotropy in singularities also requires a stable numerical method on anisotropic meshes and a metric gradation to regularize and control the variation of the prescribed metric field. Indeed, metric fields are computed from a numerical solution which may contain steep gradients and discontinuities. Hence, non-smooth metric fields can be obtained.

Early asymptotic convergence. If the previous features of unstructured mesh adaptation are quite classical, we insist here that mesh adaptation has further consequences impacting directly numerical schemes used to approximate the flow. Indeed, a loss of convergence order generally occurs due to the presence of steep gradients (Naviers Stokes equations) or genuine discontinuities (Euler equations) in the flow, despite the use of a provably spatially high order method. The computed mesh convergence order on uniformly refined meshes is not the theoretical expected one. In the previous section, we have demonstrated that an asymptotic second order mesh convergence is easily obtained for smooth functions thanks to mesh adaptation in L^p norm. Here, we show that the theoretical convergence order of numerical schemes (here a modern second-order shock capturing solver) is also recovered thanks to anisotropic mesh adaptation, even if discontinuities are present in the flow field. Verifying numerically this second order of convergence can be considered as a first assessment of computations.

4.2. Mesh adaptation algorithm for numerical simulations

Anisotropic mesh adaptation is a non-linear problem, therefore an iterative procedure is required to solve this problem. For stationary simulations, an adaptive computation is carried out *via* a mesh adaptation loop inside which an algorithmic convergence of the mesh-solution couple is sought. This mesh adaptation loop is schematized in Algorithm 2 where $\mathcal{H}$, $\mathcal{S}$ and $\mathcal{M}$ denote respectively meshes, solutions and metrics.

Algorithm 2 Mesh Adaptation Loop for Steady Flows

Initial mesh and solution $(\mathcal{H}_0, \mathcal{S}_0^0)$ and set targeted complexity N
For $i = 0, n_{adap}$

1. $(\mathcal{S}_i)$ = Compute solution with the flow solver from pair $(\mathcal{H}_i, \mathcal{S}_i^0)$;

   If $i = n_{adap}$ break;
2. $(\mathcal{M}_{L^p,i})$ = Compute metric $\mathcal{M}_{L^p}$ according to selected error estimate from $(\mathcal{H}_i, \mathcal{S}_i)$;
3. $(\widetilde{\mathcal{M}}_{L^p,i})$ = Metric field gradation $(\mathcal{M}_{L^p,i})$;
4. $(\mathcal{H}_{i+1})$ = Generate a new adapted mesh from pair $(\mathcal{H}_i, \widetilde{\mathcal{M}}_{L^p,i})$;
5. $(\mathcal{S}_{i+1}^0)$ = Interpolate new initial solution from $(\mathcal{H}_{i+1}, \mathcal{H}_i, \mathcal{S}_i)$;

EndFor

The metric field gradation (step 3) is discussed in [61] and the interpolation stage (step 5) is detailed in [74, 75]. Steps 1, 2 and 4 are described below.

The flow solver `Wolf`. All the numerical simulations of this review article are concerned with the compressible Euler equations. Results on Navier-Stokes equation such as [76] are not discussed. Assuming that the gas is perfect, inviscid and that there is no thermal diffusion, the 3D compressible Euler equations for mass, momentum and energy conservation read:

$$\frac{\partial W}{\partial t} + \nabla \cdot \mathcal{F}(W) = 0\,, \tag{11}$$

where W is the vector of conservative variables and $\mathcal{F}$ is the convection operator $\mathcal{F}(W) = (\mathcal{F}_1(W), \mathcal{F}_2(W), \mathcal{F}_3(W))$ with:

$$W = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{pmatrix}, \mathcal{F}_1(W) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ (\rho E + p)u \end{pmatrix}, \mathcal{F}_2(W) = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vw \\ (\rho E + p)v \end{pmatrix}, \mathcal{F}_3(W) = \begin{pmatrix} \rho w \\ \rho uw \\ \rho vw \\ \rho w^2 + p \\ (\rho E + p)w \end{pmatrix}.$$

We have denoted by ρ the density, $\mathbf{u} = (u, v, w)$ the Cartesian velocity vector, $E = T + \frac{\|\mathbf{u}\|^2}{2}$ the total energy and $p = (\gamma - 1)\rho T$ the pressure with $\gamma = 1.4$ the ratio of specific heats and T the temperature.

To discretize convective terms, `Wolf` uses the mixed-element-volume (MEV) approach initiated by Dervieux et al. in numerous papers [77, 78, 79, 80]. It is a vertex-centered finite volume scheme applied to tetrahedral unstructured meshes. This scheme uses a particular edge-based formulation with upwind elements. Detailed descriptions are given in [45].

This flow solver proposes several approximate Riemann solvers to compute numerical fluxes, among which Roe [81] and HLLC [82] solvers. The spatial high-order scheme is derived according to a MUSCL (Monotone Upwind Schemes for Conservation Laws) type method using downstream and upstream tetrahedra [79]. This approach is compatible with vertex-centered and edge-based formulations. Consequently, an inexpensive high-order extension of monotone upwind schemes is easily derived. The flux integration based on the edges and their corresponding upwind elements is a key-feature in order to preserve the positivity of the density for vertex-centered formulation. The MUSCL type method is combined with the Dervieux limiter [80], which is a generalization of the Superbee limiter with three entries, to guarantee the TVD (Total Variation Diminishing) property of the scheme. Several boundary conditions are available: body slip, inflow, outflow, free-stream, engine, transmitting, ... They are generally weakly imposed by solving a Riemann problem on the boundary.

As regards time advancing, several temporal scheme have been implemented. Explicit time stepping algorithm can be used by means of a strong-stability-preserving (SSP) Runge-Kutta schemes [83, 84]. Explicit temporal schemes go up to order 4. Implicit schemes are also available, such as standard backward differencing method BDF1 and BDF2 with or without Defect Correction [85] or matrix-free approach coupling the GMRES algorithm with a LU-SGS pre-conditioner [86].

To improve `Wolf`'s efficiency, a mesh entities renumbering based on space-filling curves has been implemented together with a shared-memory parallelization based on the pthreads paradigm [87].

One of the main specificities of the selected methodology is how the numerical dissipation is applied to stabilize the numerical scheme. This approach is well-suited for anisotropic adapted meshes. More precisely, the numerical dissipation is directional (edge by edge) and uses the mesh topology. This means that, for each vertex, each edge numerical flux computes its own directional dissipation. In this scheme, the stabilization term varies directionally, as compared to standard methods where the dissipation at each vertex is the same in all directions. This leads to a less dissipative second order scheme. Moreover, the dissipation term is computed according to the mesh topology, thus no parameter needs to be tuned. We observe that anisotropic meshes enhance considerably the numerical method because it provides the optimal dissipation in each direction.

The local remesher `Feflo.a`. Generating a valid tetrahedral mesh for a given domain of $\mathbb{R}^3$ of arbitrary complexity was the main problem at the end of the 90's. Different methods have demonstrated a good efficiency and reliability: advancing front method [6, 88], constrained global Delaunay [89, 90, 91], or a combination of both [92]. Based on the unit-mesh concept (Relations (1-2)), their extension to generate 3D anisotropic mesh may seem straightforward. Indeed, from a theoretical point of view, it consists in computing all relevant geometric information with respect to the provided metric field: volume, length, angle, or the Delaunay cavity. However, from a practical point of view, their implementation in 3D is far more complex and tedious due to the huge variation in terms of size and orientation (as required in an anisotropic context) which are difficult to handle properly for all the aforementioned methods. When dealing with anisotropic surface meshes, the frontal methods generally do not succeed to close the front, while the Delaunay-based will fail during the boundary recovery phase. Consequently, being able to certify that during an adaptive procedure that any of these methods will succeed to produce an adapted mesh is not at all guaranteed.

Consequently, it is of great interest to develop techniques that modify a mesh locally [29, 58, 42, 55, 54] while always keeping a valid 3D mesh. Several techniques have been developed in that direction.

The extension of the Delaunay kernel to follow an anisotropic metric is studied in [58, 93]. The main difficulty is to take into account the variation of the metric in the domain. In that case, only the volume mesh is adapted and the surface mesh is kept as is and adapted independently, thus limiting its use when strong anisotropic phenomena impact the boundary. Another set of methods consists of using several simple local operators: point insertion on edge, point collapse, and edge and face swapping. If these standard operators are monitored by an adequate metric-based quality function, then unit-meshes are generated. One main advantage is that a very high level of anisotropy is reached $O(1:10^6)$ [55, 94] and that the surface mesh is adapted conjointly with the volume mesh [95, 54]. A typical quality function gathers the control of length and volume in the metric:

$$Q_{\mathcal{M}}(K) = \frac{36}{3^{\frac{1}{3}}} \frac{|K|_{\mathcal{M}}^{\frac{2}{3}}}{\sum_{i=1}^{6} \ell_{\mathcal{M}}^2(\mathbf{e}_i)} \in [0,1], \tag{12}$$

where e_i are the edges of element K. Elements having a quality greater than 0.8 are declared to be unit with respect to $\mathcal{M}$. If the previous operators are well suited to generate anisotropic meshes, they do not encompass all the kinds of meshes required in CFD like hybrid entities or quasi-structured meshes for boundary layers. In addition, adding one feature to the local remesher requires an update to all the local operators. This makes the code more complicated to maintain or may impact its robustness unfavorably. To this end, a new cavity-based operator was introduced in [96]. The idea is to use a unique operator. Only its initialization defines the scope of the underlying local mesh modification. Each operator is equivalent to a node insertion or reinsertion. This operator consists in the definition of a cavity $\mathcal{C}$ (set of elements to be removed) and a reconnection $\mathcal{R}$ (set of new elements to be added), so that the updated mesh $\mathcal{H}^{k+1}$ writes:

$$\mathcal{H}^{k+1} \equiv \mathcal{H}^k - \mathcal{C} + \mathcal{R}$$

In Figure 8, several choices of $\mathcal{C}$ and $\mathcal{R}$ are given in order to recover the standard simple operators: a point insertion or a collapse or an edge swap. Starting from an initial choice of $\mathcal{C}$, the validity of $\mathcal{H}^{k+1}$ is monitored through cavity optimizations (enlargement or reduction). In that case, one cavity operation is equivalent to the combination of multiple simple operators at once (collapse and swap, many swaps, split and collapse, etc.). This operator was derived also to adapt a surface mesh, generate a boundary layer mesh with hybrid entities [94] and to generate quasi-structured meshes [97, 98].

4.3. Feature-based anisotropic mesh adaptation

Two major difficulties occurring when applying mesh adaptation to numerical simulations are:

- the solution of the problem is not known, only the numerical approximation is available
- a control of the approximation error is expected.

We demonstrate how this problem can be simplified to the specification of a mesh that is optimal for the interpolation error.

4.3.1. Controlling the approximation error

We describe how the interpolation theory is applied when only u_h, a piecewise linear approximation of the solution, is known. Indeed, in this particular case, the interpolation error estimate is not applied directly to u nor to u_h.

Let $\bar{V}_h^k$ be the space of piecewise polynomials of degree k (possibly discontinuous) and V_h^k be the space of continuous piecewise polynomials of degree k associated with a given mesh $\mathcal{H}$ of domain Ω_h. We denote by R_h a reconstruction operator applied to numerical approximation u_h. This reconstruction operator can be either a recovery process [99], a hierarchical basis [100], or an operator connected to an a posteriori estimate [101]. We assume that the reconstruction $R_h u_h$ is better than u_h for a given norm $\|.\|$ in the sense that:

$$\|u - R_h u_h\| \leq \alpha \|u - u_h\| \quad \text{where } 0 \leq \alpha < 1\,.$$

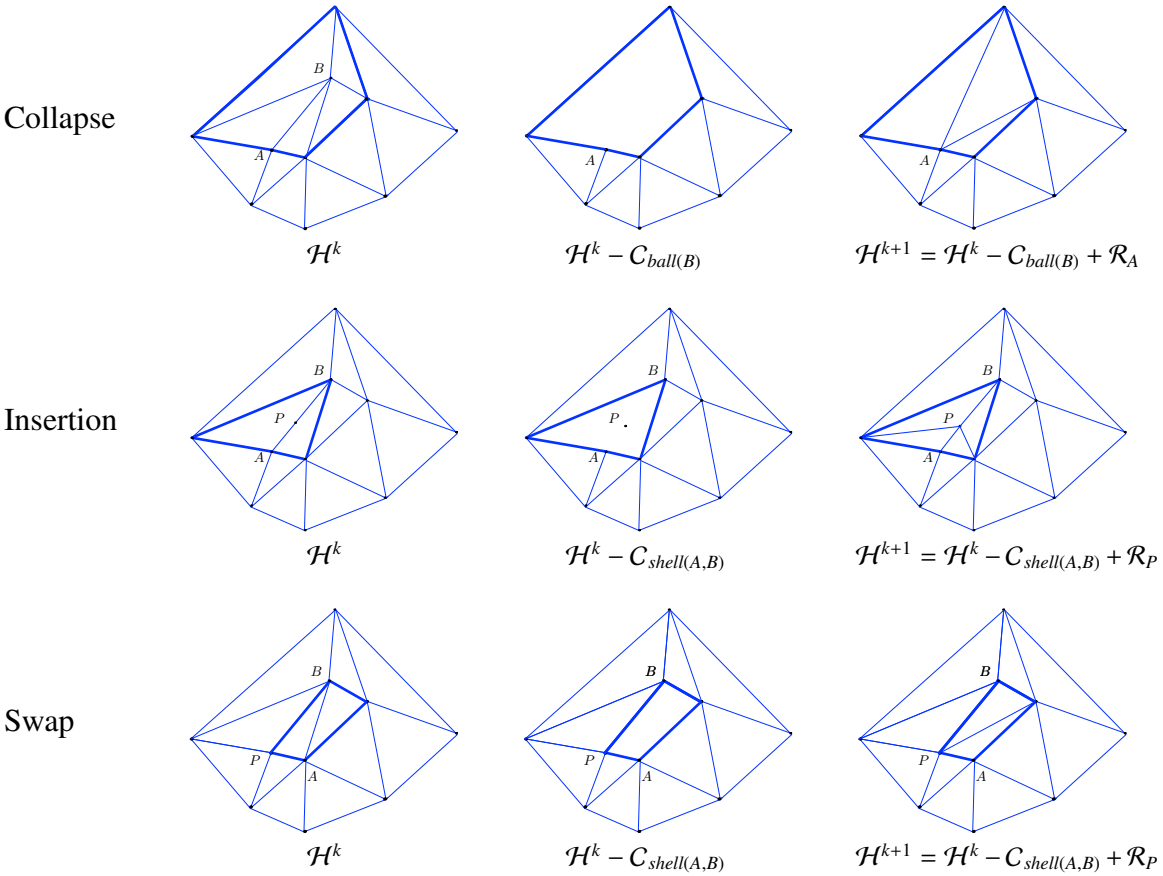


Figure 8: Each meshing operator is equivalent to a node (re)insertion.

From the triangle inequality, we deduce:

$$\|u - u_h\| \leq \frac{1}{1-\alpha}\|R_h u_h - u_h\|\,.$$

If reconstruction operator R_h has the property: $\Pi_h R_h \phi_h = \phi_h\,,\ \forall \phi_h \in V_h^1$, the approximation error of the solution can be bounded by the interpolation error of reconstructed function $R_h u_h$:

$$\|u - u_h\| \leq \frac{1}{1-\alpha}\|R_h u_h - \Pi_h R_h u_h\|\,.$$

From previous section, if $\mathcal{H}_{L^p}$ is an optimal mesh to control the interpolation error in L^p norm of $R_h u_h$, then the following upper bound of the approximation error can be exhibited:

$$\|u - u_h\|_{L^p(\Omega_h)} \leq \frac{3\,N^{-\frac{2}{3}}}{1-\alpha}\left(\int_\Omega \det\left(|H_{R_h u_h}(\mathbf{x})|\right)^{\frac{p}{2p+3}}\,\mathrm{d}\mathbf{x}\right)^{\frac{2p+3}{3p}}\,.$$

Remark 4.1. *It is important to note that $\mathcal{M}_{L^p}$ defined by Relation (5) applied to $R_h u_h$ does not allow us to generate an optimal adapted mesh to control the approximation error $\|u - u_h\|$. If all assumptions are verified, this analysis states that such generated adapted meshes controls (we have only an upper bound) the approximation error.*

In the context of numerical simulations, u_h lies in V_h^1 and its derivatives ∇u_h in $\bar{V}_h^0$. We propose a reconstruction operator from V_h^1 into V_h^2 based on $\mathbb{P}_2$ Lagrange finite element test functions. As approximate solution u_h is only

known at mesh vertices, we need to reconstruct mid-edge values. To this end, we consider the L^2-projection operator $\mathcal{P} : \bar{V}_h^0 \to V_h^1$ defined by [102]:

$$\nabla_R u_h = \mathcal{P}(\nabla u_h) = \sum_{\mathbf{p}_i \in \mathcal{H}} \nabla_R u_h(\mathbf{p}_i)\phi_i \quad \text{where} \quad \nabla_R u_h(\mathbf{p}_i) = \frac{\sum_{K_j \in S_i} |K_j| \nabla(u_{h|K_j})}{\sum_{K_j \in S_i} |K_j|} ,$$

where $\mathbf{p}_i$ denotes the i^{th} vertex of mesh $\mathcal{H}$, S_i is the stencil of $\mathbf{p}_i$, ϕ the basis function of V_h^1 and $|K_j|$ denotes the volume of element K_j. These nodal recovered gradients are used to evaluate mid-edge values. For edge $\mathbf{e} = \mathbf{pq}$, the mid-edge value $u_h(\mathbf{e})$ is given by:

$$u_h(\mathbf{e}) = \frac{u_h(\mathbf{p}) + u_h(\mathbf{q})}{2} + \frac{\nabla_R u_h(\mathbf{p}) - \nabla_R u_h(\mathbf{q})}{8} \cdot \mathbf{pq} ,$$

which corresponds to a cubic reconstruction. The reconstructed function $R_h u_h$ of V_h^2 writes:

$$R_h u_h = \sum_{\mathbf{p}_i} u_h(\mathbf{p}_i)\psi_{\mathbf{p}_i} + \sum_{\mathbf{e}_j} u_h(\mathbf{e}_j)\psi_{\mathbf{e}_j} ,$$

where $\psi_{\mathbf{p}} = \phi_{\mathbf{p}}(2\,\phi_{\mathbf{p}} - 1)$ and $\psi_{\mathbf{e}} = 4\,\phi_{\mathbf{p}}\,\phi_{\mathbf{q}}$ are the $\mathbb{P}_2$ Lagrange test functions. This reconstructed function can be rewritten $R_h u_h = u_h + z_h$ and by definition verifies:

$$\Pi_h R_h u_h = u_h \quad \text{thus} \quad \Pi_h z_h = 0 .$$

Therefore, we deduce:

$$\|R_h u_h - \Pi_h R_h u_h\| = \|u_h + z_h - u_h\| = \|z_h - \Pi_h z_h\| .$$

Finally, the approximation error can be estimated by evaluating the interpolation error of z_h:

$$\|u - u_h\| \;\leq\; \frac{1}{1-\alpha}\|z_h - \Pi_h z_h\| \;\leq\; \frac{3\,N^{-\frac{2}{3}}}{1-\alpha}\left(\int_\Omega \det\left(|H_{z_h}(\mathbf{x})|\right)^{\frac{p}{2p+3}} \mathrm{d}\mathbf{x}\right)^{\frac{2p+3}{3p}} .$$

Note that the Hessian of z_h lies in $\bar{V}_h^0$. If nodal values are needed to build $\mathcal{M}_{L^p}$, then the L^2-projection operator can be applied to these Hessians [102]. This recovery procedure is similar to the ones of [99, 103]. Other reconstruction operators can be applied such as the double L^2-projection, the least square method, or eventually the Green formula based approach [60].

Some conclusions. Based on previous theoretical results, Theorem 3.3, and what has been observed on analytical functions in Section 3.4, we may expect the following properties for mesh adaptive numerical simulations:

- highly anisotropic adapted meshes
- ability to capture all scales of the flow, *i.e.,* being able to capture at the same time strong phenomena (shocks, boundary layer, etc.) and weak ones (vortices, contact discontinuities, etc.)
- an asymptotic second order convergence for smooth flows or flows with steep gradients
- a higher order convergence for flows with genuine discontinuities, at best second order.

Moreover, the proposed error estimate is independent of the considered problem, *i.e.,* the PDE. Indeed, it is geometric. Therefore, it can be applied to a large spectrum of applications.

The achievement of second order convergence for discontinuous flows has been presented in [65] and the comparison with experiments in [45]. Here, we present a real-life sonic boom application to show the efficiency of the proposed methodology.

4.3.2. A low boom supersonic aircraft

We simulate a supersonic flow of a low-boom-shaped supersonic business jet geometry (SSBJ) provided by Dassault-Aviation during the HISAC european project [104]. This is a complex geometry with an integrated engine over the fuselage to minimize the impact of the nacelles on the sonic boom. Moreover, the wing has a double dihedral angle. The first dihedral angle is at the junction of the wing and the fuselage. The second one is where the wing swept angle changes. The aircraft length is 42 meters and it has a wing span of 20 meters. The surface mesh size on the aircraft geometry ranges between 0.2 mm and 12 cm, Figure 9. This already represents a size variation of five orders of magnitude with respect to the aircraft size. The SSBJ geometry is considered inside a cylindrical computational domain of 2.25 km length and 1.5 km diameter. This represents a scale factor of 10^7 if the size of the domain is compared to the maximal accuracy of the low boom jet surface mesh.

The jet is flying at cruise with Mach number 1.6, an angle of attack of 3°, and an altitude of 45,000 feet. Such supersonic flows involve highly anisotropic physical features. Indeed, as for a body flying at a supersonic speed, each geometric singularity generates a cone-shaped shock wave ; a multitude of conic shock waves are emitted by the aircraft geometry. They generally coalesce around the aircraft and propagate to the ground.

In regards to mesh adaptation, we choose to control the interpolation error on the local Mach number in L^2 norm. 32 mesh adaptation iterations are performed. We split the adaptation loop in 4 steps with an increasing complexity specification at each step. Within each step, an adapted mesh of fixed complexity is converged. Final step meshes are used for the computation of the global mesh convergence order. Starting from a coarse uniform mesh containing 772,572 vertices and 3,768,534 tetrahedra, the final adapted mesh contains 9.1 million vertices and 53.9 million tetrahedra. This mesh is illustrated in Figure 11. The obtained adapted meshes are highly anisotropic. For the last one, the mean anisotropic ratio is 372 and the mean anisotropic quotient is 49,051. The last quantity signifies that the anisotropy leads to a mesh complexity reduction by more than four orders of magnitude as compared to an isotropic adapted mesh.

Such adapted meshes considerably enhance the efficiency of the flow solver. We first notice the multitude of shock waves (Mach cones) that have been accurately computed in the SSBJ near-field, Figure 11 (right). All the details and scale of the solution have been captured and are represented inside the mesh. Second, mesh refinements along Mach cones have been propagated in the whole computational domain with high accuracy (small size elements) allowing the flow solver to achieve an accurate flow prediction everywhere, Figure 10 (left) and 11 (left). This result points out that the numerical dissipation of the flow solver is drastically reduced with the aid of anisotropic mesh refinement. Finally, we observe that the global spatial second order of convergence is asymptotically reached for the local Mach number on the sequence of adapted meshes, see Figure 10 (right).

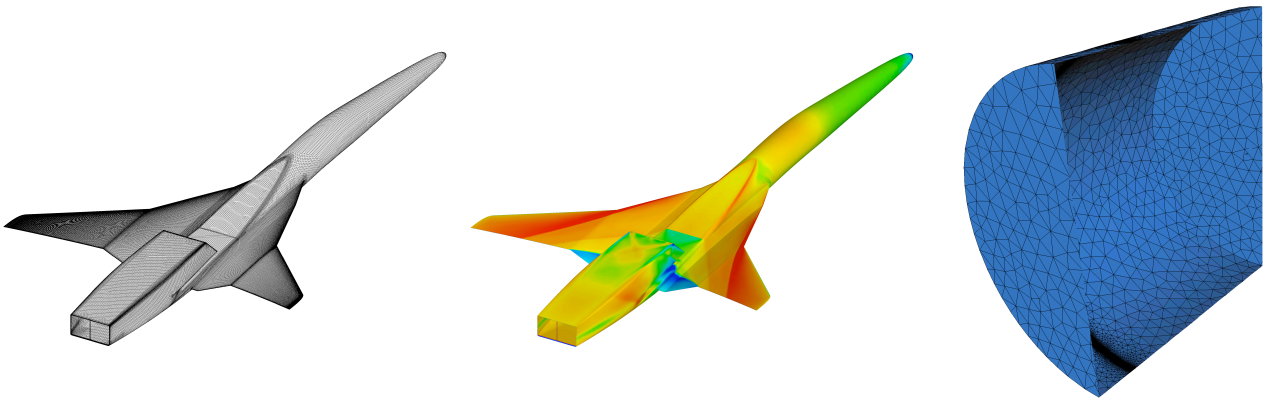


Figure 9: Supersonic SSBJ. Left and middle, surface mesh and solution (local Mach number) of the low-boom-shaped supersonic business jet geometry. Right, the 2.5 kilometers cylindrical computational domain: the SSBJ is represented by a little dot !

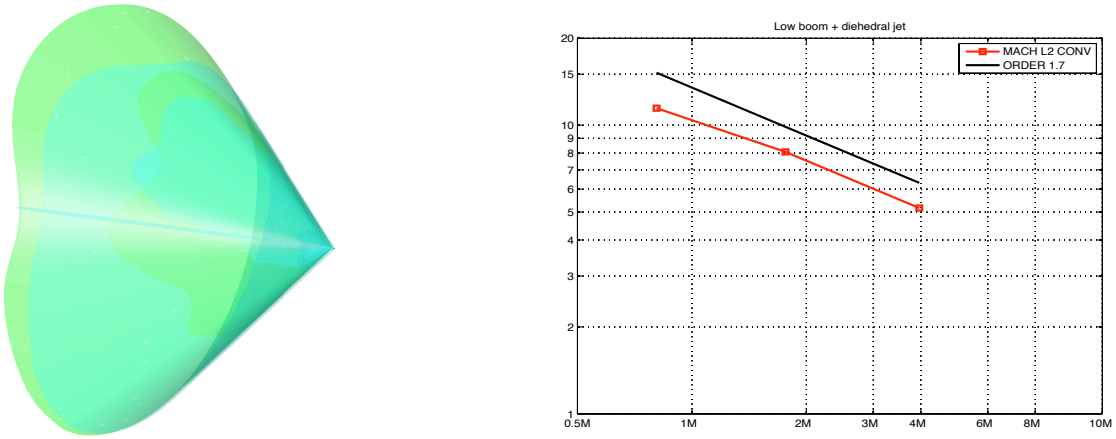


Figure 10: Supersonic SSBJ. Left, Mach cone emitted by the SSBJ. The maximal Mach cone diameter is 1.25 km. The solution is accurately propagated in the whole computational domain. Right, global L^2 norm spatial convergence for the local Mach number on a sequence of adapted meshes.

4.3.3. Chronology of the results

Figure 12 displays the evolution of the results on the SSBJ simulation. To evaluate the efficiency of the adaptive method, we analyze the distance of of the propagation of the shock wave from the aircraft: the further away the better. It quantifies the reduction of the numerical dissipation of the flow solver. This distance is evaluated by the ratio R/L where L is the aircraft length and R the distance of propagation. Left, an isotropic mesh adaptation in L^∞ norm done in 2003 where $R/L = 1.25$ is reached [105]. Middle, an anisotropic mesh adaptation in L^∞ norm done in 2006 where $R/L = 3$ is reached [106]. Right, an anisotropic mesh adaptation in L^p norm done in 2007 where $R/L = 40$ is reached [65].

4.4. Goal-oriented anisotropic mesh adaptation

In the previous sections, metric-based anisotropic mesh adaptation methods are limited to the minimization of some interpolation errors for some solution fields, the "sensors", and did not take into account the PDE under consideration. If, for many applications, this simplifying standpoint is an advantage, there are also many applications where Hessian-based adaptation is far from optimal regarding the way the degrees of freedom are distributed in the computational domain. Indeed, Hessian-based methods aim at controlling the interpolation error but this purpose is not often so close to the objective that consists in obtaining the best solution of a PDE. This is particularly true in many engineering applications where a specific function needs to be accurately evaluated: lift, drag, heat flux, pressure field, etc. Hessian-based adaptation has not been especially designed to address this issue.

In contrast with the previous set of methods, the objective of *goal-oriented* mesh adaptation is to derive the optimal mesh to observe a given scalar-output functional j depending on solution w. To this end, the control of the approximation error on the output functional, $j(w) - j(w_h)$, is examined. Goal-oriented methods result from a series of papers dealing with *a posteriori estimates* (e.g. [107, 108, 72, 70, 41, 109, 71]). But, extracting information concerning anisotropy from an *a posteriori* estimate is a difficult task.

In our point of view, metric analysis and goal-oriented analysis are complementary. Indeed, a metric-based method specifies the object of our search through an accurate description of the ideal mesh while a goal-oriented method specifies precisely the purpose of the search in terms of which error will be reduced. It is then very motivating to seek a combination of both methods, with the hope of obtaining a metric-based specification of the best mesh to reduce the error committed on a target output functional. This will be achieved from some considerations: a reliable continuous mesh model, an *a priori* estimate, and specific numerical schemes allowing accurate approximations of $\Psi_h(w) - \Psi(w)$. These three items differ from classical approaches of goal-oriented mesh adaptation.

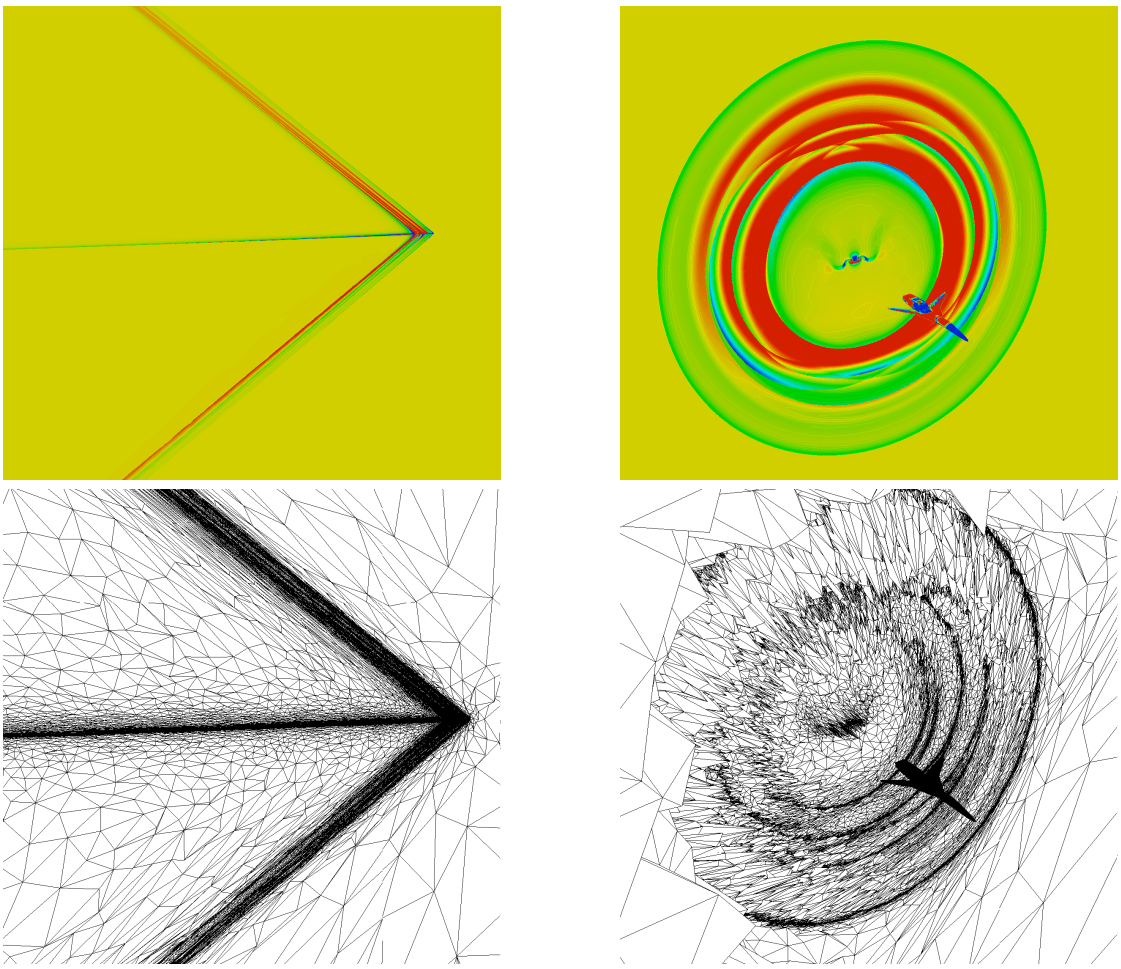


Figure 11: Supersonic SSBJ. Views of the local mach number iso-values (top) and the final anisotropic adapted mesh composed of 9.1 million vertices and 53.9 million tetrahedra (bottom) in the symmetry plane (left) and in a plane behind the aircraft orthogonal to the jet path (right).

4.4.1. A more accurate non-linear error analysis

Assumptions and definitions. Let V be a space of functions (at least a Banach space). We write the state equation under a variational statement:

$$w \in V \ , \ \forall \varphi \in V \ , \ (\Psi(w), \varphi) = 0, \tag{13}$$

where operator $(,)$ holds for a $V' \times V$ product, V' is the topological dual of V and w is the solution of this equation. Symbol Ψ holds for a functional that is linear with respect to test function φ but *a priori* non linear with respect to w. Let V_h be a subspace of $\mathcal{V} = V \cap C^0$ of finite dimension N. The discrete state equation is written as follows:

$$w_h \in V_h \ , \ \forall \varphi_h \in V_h \ , \ (\Psi_h(w_h), \varphi_h) = 0. \tag{14}$$

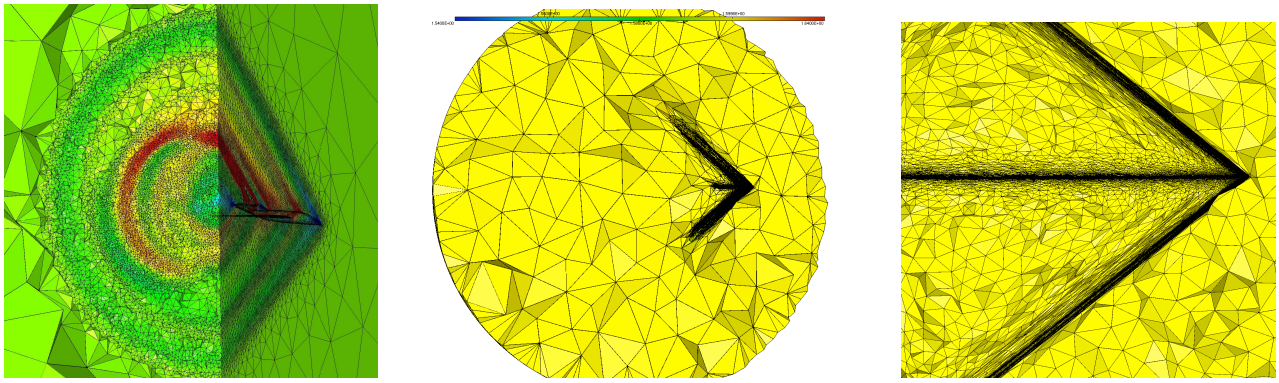
For a solution w of state system (13), we define a *functional output* as:

$$j \in \mathbb{R}; \quad j(w) = (g, w),$$

where g is a function of V. We introduce the continuous adjoint w^*, solution of the system:

$$w^* \in V \ , \ \forall \psi \in V \ , \ \left(\frac{\partial \Psi}{\partial w}(w)\psi, w^* \right) = (g, \psi), \tag{15}$$

Isotropic L^∞ error estimate [105] | Anisotropic L^∞ error estimate [106] | Anisotropic L^p error estimate [65]



798,756 vertices
4,714,162 tetrahedra
$R/L = 1.25$ (50 m)

1,775,049 vertices
10,474,598 tetrahedra
$R/L = 3$ (120 m)

3,764,591 vertices
22,324,258 tetrahedra
$R/L = 40$ (1.6 km)

Figure 12: Supersonic SSBJ. Progress on the SSBJ simulation.

and the discrete adjoint state w_h^* which is solution of:

$$w_h^* \in V_h \ , \ \forall \psi_h \in V_h \ , \ \left(\frac{\partial \Psi_h}{\partial w}(\Pi_h w)\psi_h, w_h^* \right) = (g, \psi_h). \tag{16}$$

A priori estimation. The objective is to estimate the approximation error made on the output functional:

$$\delta j = j(w) - j(w_h) \, ,$$

as a function of continuous solutions, of continuous residuals, and of discrete residuals. For the *a priori* analysis, we assume that solutions w and w^* are sufficiently regular: $w \in \mathcal{V}$ and $w^* \in \mathcal{V}$, and that we have an interpolation operator: $\Pi_h : \mathcal{V} \rightarrow V_h$. Error δj is split as follows:

$$\delta j = j(w) - j(w_h) = (g, w - \Pi_h w) + (g, \Pi_h w - w_h).$$

δj is now composed of an *interpolation error* and of an *implicit error* which involves only discrete terms. The implicit error term has been thoroughly analyzed in [43], and can be approximated as:

$$(g, \Pi_h w - w_h) \ \approx \ \left(\frac{\partial \Psi}{\partial w}(w)(\Pi_h w - w), w^* \right) + ((\Psi_h - \Psi)(w), w^*) \, .$$

Thus, thanks to the continuous adjoint of Equation (15), we finally obtain:

$$\delta j \quad \approx \quad ((\Psi_h - \Psi)(w), w^*) \, . \tag{17}$$

It remains to precise the context of the study and to exhibit from (17) a formulation specifying the optimal mesh.

The case of the steady compressible Euler equations. We study how Equation (17) can be applied in the context of the steady compressible Euler equations. In [43], we restrict ourselves to a particular discretization of these equations (equivalent to the one presented in Section 4.2) and we consider a variational analysis. If we neglect the boundary terms, the final estimate reads:

$$|j(W) - j(W_h)| \quad \approx \quad \| \, |\nabla W^*| \cdot |\mathcal{F}(W) - \Pi_h \mathcal{F}(W)| \, \|_{L^1(\Omega_h)} \tag{18}$$

We observe that this estimate of δj is expressed in term of the interpolation error in L^1 norm of the Euler fluxes applied to the continuous solution W weighted by the gradient of the continuous adjoint state W^*.

4.4.2. Error model minimization

In order to derive an optimal mesh for the observed output functional, we work in the continuous mesh framework which is made easier thanks to the *a priori* estimate. Estimate (18) is rewritten in a continuous form (boundary terms are still neglected to simplify the analysis, see [43] for their analysis and a discussion of their influence):

$$(g, W_h - W) \approx E_{go}(\mathbf{M}) = \int_\Omega |\nabla W^*| \cdot |\mathcal{F}(W) - \pi_{\mathcal{M}}\mathcal{F}(W)| \, \mathrm{d}\Omega \,, \tag{19}$$

where $\mathbf{M} = (\mathcal{M}(\mathbf{x}))_{\mathbf{x}\in\Omega}$ is a continuous mesh and $\pi_{\mathcal{M}}$ is the continuous linear interpolate. We are now focusing on the following (continuous) mesh optimization problem:

$$\text{Find } \mathbf{M}_{go} = \text{Argmin}_{\mathbf{M}} \, E_{go}(\mathbf{M}), \quad \text{such that } \; C(\mathbf{M}) = N \,. \tag{20}$$

Then, introducing the continuous interpolation error, we can write the simplified error model as follows:

$$\mathbf{E}(\mathbf{M}) \;=\; \int_\Omega \text{trace}\left(\mathcal{M}^{-\frac{1}{2}}(\mathbf{x})\, \mathbf{H}(\mathbf{x})\, \mathcal{M}^{-\frac{1}{2}}(\mathbf{x})\right) \mathrm{d}\mathbf{x}$$

$$\text{in which} \quad \mathbf{H}(\mathbf{x}) = \sum_{j=1}^{5} \left([\Delta x]_j(\mathbf{x}) + [\Delta y]_j(\mathbf{x}) + [\Delta z]_j(\mathbf{x})\right), \tag{21}$$

with $[\Delta x]_j = \left|\dfrac{\partial W_j^*}{\partial x}\right| \left|H(\mathcal{F}_1(W_j))\right|$, $[\Delta y]_j = \left|\dfrac{\partial W_j^*}{\partial y}\right| \left|H(\mathcal{F}_2(W_j))\right|$, $[\Delta z]_j = \left|\dfrac{\partial W_j^*}{\partial z}\right| \left|H(\mathcal{F}_3(W_j))\right|$. Here, W_j^* denotes the j^{th} component of adjoint vector W^* and $H(\mathcal{F}_i(W_j))$ the Hessian of the j^{th} component of vector $\mathcal{F}_i(W)$. Solving the optimality condition provides the *optimal goal-oriented continuous mesh* $\mathbf{M}_{go} = \left(\mathcal{M}_{go}(\mathbf{x})\right)_{\mathbf{x}\in\Omega}$:

$$\mathcal{M}_{go}(\mathbf{x}) \;=\; \mathcal{M}_{L^1}(\mathbf{H}(\mathbf{x})) \;=\; N^{\frac{2}{3}} \left(\int_\Omega (\det \mathbf{H}(\bar{\mathbf{x}}))^{\frac{1}{5}} \mathrm{d}\bar{\mathbf{x}}\right)^{-\frac{2}{3}} (\det \mathbf{H}(\mathbf{x}))^{-\frac{1}{5}} \; \mathbf{H}(\mathbf{x}). \tag{22}$$

4.4.3. A supersonic business jet

Comparison with experiments and numerous numerical examples are presented in [65]. Here, we compare the result using the feature-based and the goal-oriented error estimates on the supersonic business jet simulation presented above.

The goal is to accurately compute the pressure signature only on a plane located 100 m below the aircraft. The observation plane γ has a length of 40 m and a width of 2 m while the wing span is about 17 m. The objective of this test case is to evaluate the ability of the adjoint to prescribe refinements only in areas that impact the observation region. The function is given by:

$$j(W) = \frac{1}{2} \int_\gamma \left(\frac{p - p_\infty}{p_\infty}\right)^2 \mathrm{d}\gamma \,.$$

For the feature-based mesh adaptation, the sensor is again the local Mach number and the interpolation error is controlled in L^2 norm.

The adaptive loop is divided into 5 steps of increasing complexity, each step is composed of 6 sub-iterations having a constant complexity for a total of 30 adaptations. Hessian-based and adjoint-based final adapted meshes are both composed of almost 800, 000 vertices. They are represented in Figure 14, where several cuts in the final adapted meshes for the adjoint-based (left) and Hessian-based (right) adaptations are displayed.

For the Hessian-based adaptation, the mesh is adapted in the whole computational domain along the Mach cones and in the wake providing an accurate solution everywhere in the domain, cf. Figure 13 (top right). However, if the aim is only to compute an accurate pressure signature on surface γ, then we clearly notice that a large amount of degrees of freedom is wasted in the upper part of the domain and in the wake where accuracy is not needed. In regards to adjoint-based adaptation, the mesh is mainly adapted below the aircraft in order to accurately capture all of the

shock waves that impact the observation plane while areas that do not impact the output functional are ignored with this new approach (cf. Figure 14).

Another point of interest, which is more technical, is that the Hessian-based adaptation norm prescribes a mesh size that depends on the shock intensity. A stronger shock is then more refined than a weaker one. In this simulation, shocks directly below the aircraft have a lower intensity than lateral or upper shocks emitted by the wings, see Figure 14 (bottom). Consequently, the adapted meshes obtained with the Hessian-based method are less accurate in regions that directly affect the observation plane than the adapted meshes obtained with the adjoint-based method. Indeed, with the adjoint-based strategy, the shock waves that have the most influence on the computation of the output functional are more refined, and this independently of their amplitudes.

This has a direct consequence on the accuracy of the observed output functional, see Figure 13 (bottom). At the same level of complexity, the pressure signature is clearly more accurate with adjoint-based mesh adaptation. It also demonstrates that the adjoint defines an optimal distribution of the degrees of freedom for the specific target.

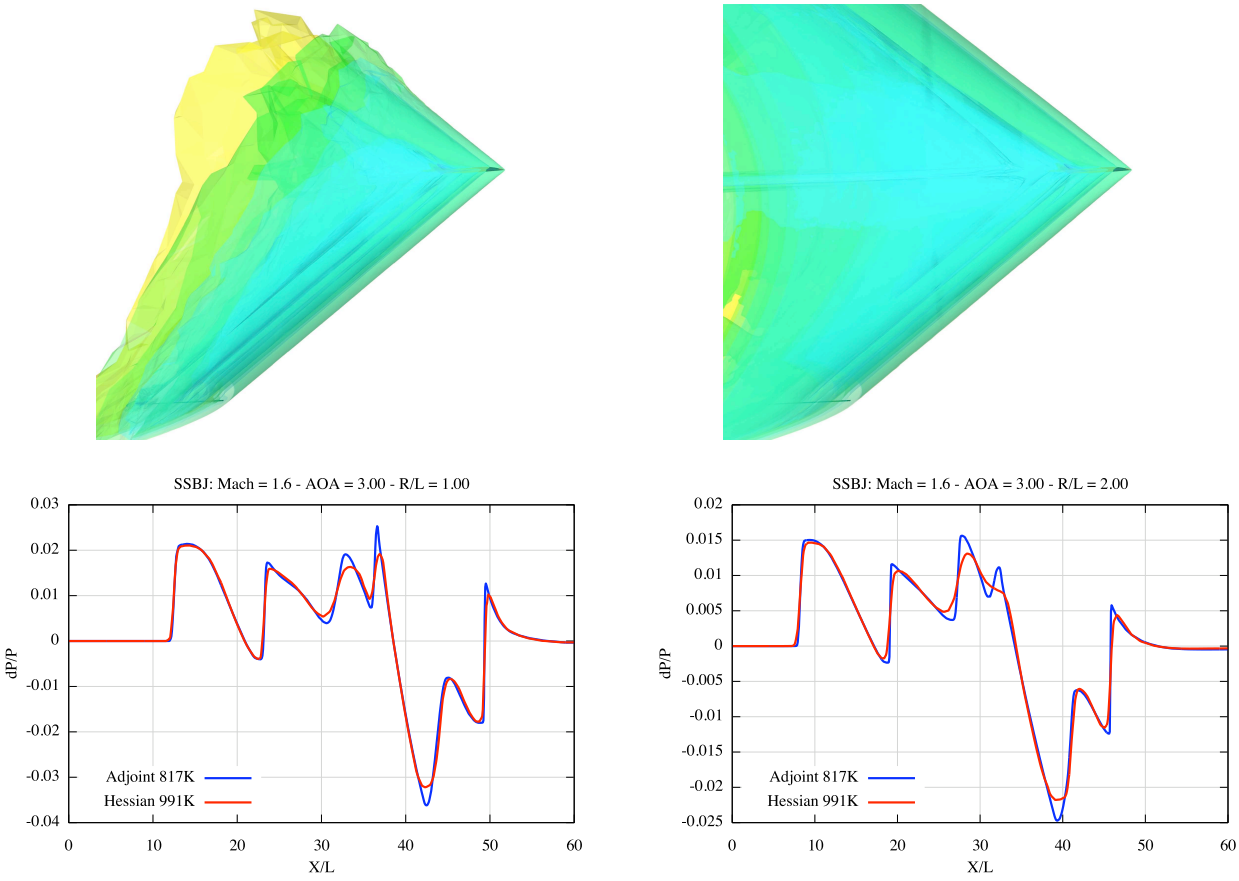


Figure 13: Top, local Mach number iso-surfaces for the adjoint-based (left) and Hessian-based (right) methods. Bottom, pressure signal extraction under the SSBJ along the x-axis at one-body (left) and two-body (right) lengths.

5. Metric-based anisotropic mesh adaptation for unsteady flows

This section presents the extension of the feature-based and goal-oriented anisotropic mesh adaptation to time-dependent simulations.

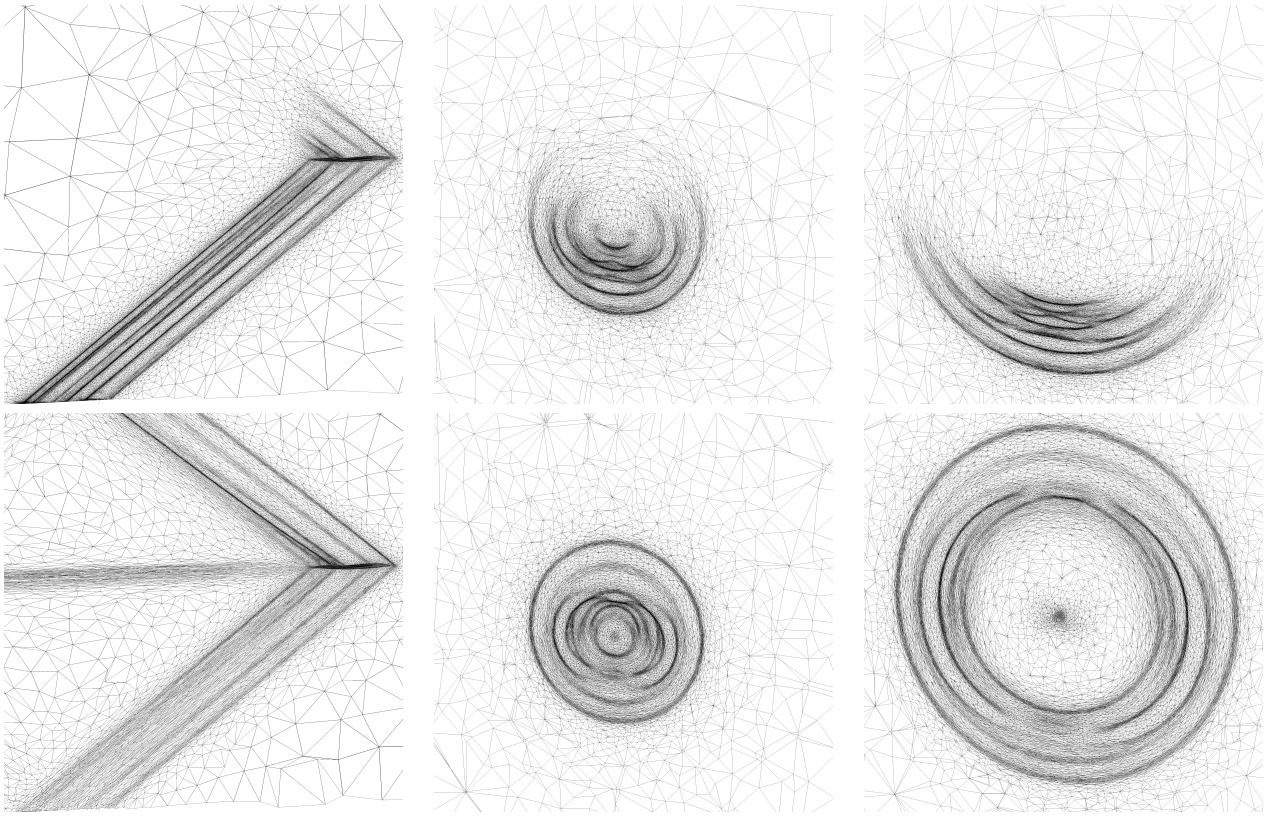


Figure 14: Cut planes through the final adapted meshes for the adjoint-based (top) and Hessian-based (bottom) methods. Left, a cut in the symmetry plane and, middle and right, two cuts with an increasing distance behind the aircraft orthogonal to its path.

5.1. Problems and motivations

The 3D simulation of unsteady flows, which is actually the typical situation for real-life problems, still remains a challenge. Indeed, these computations are very time consuming. To reduce the CPU time of these simulations while preserving their accuracy, anisotropic metric-based mesh adaptation, which has already proved its efficiency for steady problems, appears as a salutary perspective. However, its extension to the unsteady case is far from being straightforward as such simulations cumulate the difficulties arising from unsteadiness. To achieve an efficient time-accurate mesh adaptation scheme for unsteady flow, we have to overcome the following issues.

First of all, we have to remedy the problem of the lateness of the mesh with respect to the solution, *i.e.,* the mesh is lagging behind the solution in time. Indeed, if a mesh is adapted for a solution at time t, once the solution progresses in time it is clear that this mesh is no longer adapted for the subsequent time steps.

Theoretically, the previous estimates introduced in Section 4.2 need to be extended to the unsteady case. In other words, for time-dependent simulations, the temporal error is also controlled. A space-time error analysis is required.

Additionally, at each remeshing, the solution has to be interpolated on the new adapted mesh. This stage becomes crucial in the context of unsteady problems and even more if a large number of interpolations is performed, as the error due to this solution transfer accumulates throughout the simulation. Therefore, the error introduced by this stage can spoil the overall accuracy of the solution.

In regards to the meshing phase, a difficulty arises from the definition of the solver time step dt (omitting CFL factor) which is homogeneous to the smallest height of the mesh h_{min}. Consequently, a single small-height element in the whole mesh is sufficient to considerably reduce the time step and thus increase the CPU time of the simulation. This is a serious problem, especially in the context of highly anisotropic mesh adaptation, which involves highly stretched elements. The only remedy is to reduce this constraint as far as possible by generating anisotropic meshes controlling the highest h_{min} value. This implies a substantial effort on the anisotropic mesh generator as the quality of

the mesh must be perfect. For example, if the mesh generator fails to generate the minimal height element for which a size of h_{target} had been prescribed, and instead builds an element of height $h_{min} = 0.01 \times h_{target}$, then the number of solver iterations is multiplied by 100. Consequently, *not a single meshing mistake is allowed even if millions of tetrahedra are generated.*

Over the past few years, a rather large number of papers have been published dealing with mesh adaptation for *steady* numerical simulations, whereas only a small number have addressed *time-dependent* problems [59, 110, 28, 111, 50, 112, 113, 114]. All these approaches involve a large number of adaptations while introducing unquantified errors due to the transfer of the solution from the old mesh to the new one. This is especially problematic for hyperbolic problems. Moreover, they generally do not explicitly control the error made on the solution as they perform an arbitrary large and *a priori* number of adaptations. Some of them authorize the error to grow through the simulation and others cannot be extended to the case of anisotropic mesh adaptation. Finally, none of them considers the intrinsic non-linear nature of the mesh adaptation problem: the convergence of the mesh adaptation process is never addressed and therefore the obtention of the optimal mesh cannot be expected.

In order to remedy all of the problems relative to mesh adaptation for time-dependent simulations, an innovative strategy based on a fixed-point algorithm has been developed. The main goals are to guarantee a control of the spatial and temporal interpolation errors during the whole simulation and to control (reduce) the number of mesh adaptations in order to master (diminish) the error introduced by solution transfers. This new strategy starts with the observation that direct extension of steady adaptation algorithms to unsteady problems is not appropriate: specific algorithms must be developed, which truly take into account the transient nature of the solution. It relies on the assumption that the temporal error is always controlled by the spatial one, which is indeed the case when solving a linear advection problem under a CFL condition [48]. To this end, it has been proposed:

- a global fixed-point mesh adaptation algorithm to converge the mesh adaptation non-linear problem
- the extension of the multiscale error estimate to unsteady problems by proposing a L^p space-time error analysis
- the extension of the adjoint-based mesh adaptation to unsteady problems by proposing a goal-oriented space-time error analysis. This requires the computation of the unsteady adjoint state backward in time
- a conservative solution transfer to considerably diminish the error introduced by this stage of the algorithm
- a high-quality anisotropic local remeshing controlling the heights of the elements to guarantee acceptable solver time steps and to ensure maximal robustness in the meshing process. This issue has been handled with care inside `feflo.a` which is described in Section 4.2.

5.2. Matrix-free $\mathbb{P}_1$-exact conservative solution transfer

At each remeshing, the solution needs to be transferred from the previous mesh to the next one to pursue the computation. This stage becomes crucial in the context of unsteady problems and even more if a large number of transfers is performed, as the error introduced by this stage can spoil the overall accuracy of the solution. In the context of the resolution by a second order numerical scheme of a PDE system of conservation laws, as the compressible Euler system, it seems mandatory for the interpolation method to satisfy the following properties in order to obtain a consistent mesh adaptation scheme: mass conservation, $\mathbb{P}_1$ exactness preserving the second order of the adaptive strategy, and verify the maximum principle.

The mass conservation property of the interpolation operator is achieved by local mesh intersections, *i.e.,* intersections are performed at the element level. The use of mesh intersection to build a conservative interpolation process seems natural for unconnected meshes. The locality is primordial for efficiency and robustness. The idea is to find, for each element of the new mesh, its geometric intersection with all the elements of the background mesh it overlaps and to mesh this geometric intersection with simplices. We are then able to use a Gauss quadrature formula to exactly compute the mass which has been locally transferred.

High-order accuracy is obtained through the reconstruction of the gradient of the solution from the discrete data and the use of some Taylor formulae. Unfortunately, this high-order interpolation can lead to a loss of monotonicity. The maximum principle is recovered by correcting the interpolated solution in a conservative manner. Finally, the

solution values at vertices are reconstructed from this piecewise linear by element discontinuous representation of the solution. More details can be found in [74, 75].

The algorithm can be summarized as follow:

Algorithm 3 Conservative Interpolation Process

Piecewise linear (continuous or discontinuous) representation of the solution on $\mathcal{H}_{back}$

1. For all elements $K_{back} \in \mathcal{H}_{back}$, compute solution mass $m_{K_{back}}$ and gradient $\nabla_{K_{back}}$
2. For all elements $K_{new} \in \mathcal{H}_{new}$, recover solution mass $m_{K_{new}}$ and gradient $\nabla_{K_{new}}$:
   (a) compute the intersection of K_{new} with all $K^i_{back} \in \mathcal{H}_{back}$ it overlaps
   (b) mesh the intersection polygon/polyhedra of each couple of elements (K_{new}, K^i_{back})
   (c) compute $m_{K_{new}}$ and $\nabla_{K_{new}}$ using Gauss quadrature formulae

   $\Longrightarrow$ a piecewise linear discontinuous representation of the mass on $\mathcal{H}_{new}$ is obtained
3. Correct the gradient to enforce the maximum principle
4. Set the solution values to vertices by an averaging procedure.

Figure 15 points out the superiority of the $\mathbb{P}_1$-conservative solution transfer (right) with respect to the classic $\mathbb{P}_1$ interpolation (left) on an adaptive blast simulation in two dimensions. For both simulations, all parameters are the same except for the solution transfer stage. This figure shows the final solution obtained with 70 mesh adaptations, *i.e.,* a total of 70 solution transfers. The diffusion and the error introduced by the classic $\mathbb{P}_1$ solution transfer clearly spoils the solution accuracy while the solution remains very accurate with the $\mathbb{P}_1$-conservative operator.

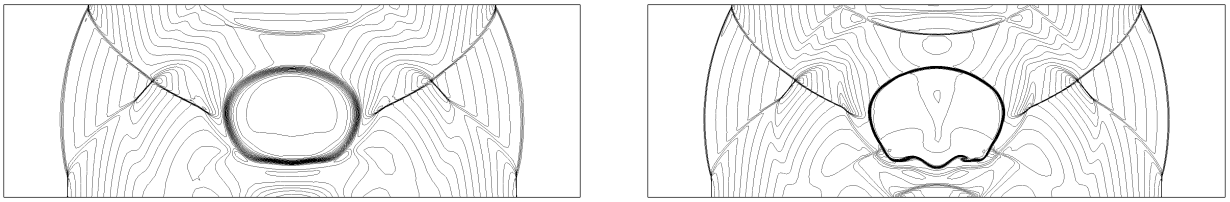


Figure 15: Final result of a blast adaptive simulation with 70 mesh adaptations. Left, using a classic $\mathbb{P}_1$ solution transfer. Right, using the $\mathbb{P}_1$-conservative solution transfer.

5.3. Feature-based unsteady anisotropic mesh adaptation

For the optimal space-time mesh, we seek to control the space-time spatial discretization error. The following assumption is made (it has been demonstrated under specific conditions in [48]): *as an explicit time scheme is used for time advancing, then the error in time is controlled by the error in space under CFL condition.* As far as the above hypothesis is true, the spatial interpolation error is a good measure of the total space-time error of the discretized unsteady system.

5.3.1. Space-time L^p error analysis

Our goal is to solve an unsteady PDE which is set in the computational space-time domain $Q = \Omega \times [0, T]$ where T is the (positive) maximal time and $\Omega \subset \mathbb{R}^3$ is the spatial domain. Let Π_h be the usual $\mathbb{P}^1$ projector. Its extension to time-dependent functions reads:

$$(\Pi_h \varphi)(t) = \Pi_h(\varphi(t)), \ \forall\, t \in [0, T].$$

The considered problem of mesh adaptation consists in finding the space-time mesh $\mathcal{H}$ of Q that minimizes the space-time linear interpolation error $u - \Pi_h u$ in L^p norm. The problem is thus stated in an *a priori* way:

$$\text{Find } \mathcal{H}_{opt} \text{ having } N_{st} \text{ space-time vertices such that } \mathbf{E}_{L^p}(\mathcal{H}_{opt}) = \min_{\mathcal{H}} \|u - \Pi_h u\|_{L^p(\Omega_h \times [0,T])}.$$

In the continuous mesh framework, we rewrite this problem under the continuous form:

$$\text{Find } \mathbf{M}_{L^p} = (\mathcal{M}_{L^p}(\mathbf{x},t))_{(\mathbf{x},t)\in\mathcal{Q}} \text{ such that } \mathbf{E}_{L^p}(\mathbf{M}_{L^p}) = \min_{\mathbf{M}} \|u - \pi_{\mathcal{M}} u\|_{L^p(\Omega\times[0,T])}\,, \tag{23}$$

under the space-time constraint:

$$C_{st}(\mathbf{M}) = \int_0^T \tau(t)^{-1} \left(\int_\Omega d_{\mathcal{M}}(\mathbf{x},t)\, \mathrm{d}\mathbf{x} \right) \mathrm{d}t = N_{st}\,. \tag{24}$$

where $\tau(t)$ is the time step used at time t of interval $[0,T]$. Introducing the continuous interpolation error, we recall that we can write the continuous error model as follows:

$$\mathbf{E}_{L^p}(\mathbf{M}) = \left(\int_0^T \int_\Omega \mathrm{trace}\left(\mathcal{M}^{-\frac{1}{2}}(\mathbf{x},t) |H_u(\mathbf{x},t)| \mathcal{M}^{-\frac{1}{2}}(\mathbf{x},t) \right)^p \mathrm{d}\mathbf{x}\, \mathrm{d}t \right)^{\frac{1}{p}}.$$

where H_u is the Hessian of sensor u. To find the optimal space-time continuous mesh, Problem (23-24) is solved in two steps. First, a spatial minimization is done for a fixed t. Second, a temporal minimization is performed. Here, we consider the case where the time step τ is specified by the user as a function of time $t \to \tau(t)$. A similar analysis can be done to deal with the case of an explicit time advancing solver subject to Courant time step condition. It leads to the expression of the optimal space-time metric $\mathbf{M}_{L^p}$ for a prescribed time step $\tau(t)$:

$$\mathcal{M}_{L^p}(\mathbf{x},t) = N_{st}^{\frac{2}{3}} \left(\int_0^T \tau(t)^{-\frac{2p}{2p+3}} \mathcal{K}(t) \mathrm{d}t \right)^{-\frac{2}{3}} \tau(t)^{\frac{2}{2p+3}} (\det |H_u(\mathbf{x},t)|)^{-\frac{1}{2p+3}} |H_u(\mathbf{x},t)|\,, \tag{25}$$

where $\mathcal{K}(t) = \left(\int_\Omega (\det |H_u(\mathbf{x},t)|)^{\frac{p}{2p+3}} \mathrm{d}\mathbf{x} \right)$. The following optimal error is finally obtained:

$$\mathbf{E}_{L^p}(\mathbf{M}_{L^p}) = 3\, N_{st}^{-\frac{2}{3}} \left(\int_0^T \tau(t)^{-\frac{2p}{2p+3}} \mathcal{K}(t) \mathrm{d}t \right)^{\frac{2p+3}{3p}}. \tag{26}$$

5.3.2. Error analysis for the global fixed-point mesh adaptation algorithm

The computation of the optimal instantaneous continuous mesh given by Relation (25) involves a global normalization term which requires the knowledge of quantities over the whole simulation time frame. Thus, the complete simulation must be performed before evaluating any space-time continuous mesh. To solve this issue, we suggest to consider a global fixed-point mesh adaptation algorithm covering the whole time frame $[0,T]$. This iterative algorithm is used to converge the non-linear mesh adaptation problem, *i.e.,* converging the mesh-solution couple. This is also a way to predict the solution evolution and to adapt the mesh accordingly.

Moreover, the previous analysis provides the optimal size of the adapted meshes for each time level. Hence, this analysis requires the mesh to be adapted at each flow solver time step which is inconceivable in practical applications. We propose to use a coarse adapted discretization of the time axis. The basic idea consists in splitting the simulation time frame $[0,\ T]$ into n_{adap} adaptation sub-intervals:

$$[0,\ T] \ = \ [0 = t_0,\ t_1] \cup \ldots \cup [t_i,\ t_{i+1}] \cup \ldots \cup [t_{n_{adap}-1},\ t_{n_{adap}} = T]\,,$$

and to keep the same adapted spatial mesh for each time sub-interval. On each sub-interval, the mesh is adapted to control the solution accuracy from t_i to t_{i+1}. Consequently, the time-dependent simulation is performed with n_{adap} different adapted meshes. This drastically reduces the number of remeshings during the simulation, hence the number of solution transfers. This provides a first answer to the adaptation of the whole space-time mesh, the spatial mesh being kept constant for each sub-interval when the global space-time mesh is visualized,

Now, we want to extend the previous analysis to the fixed-point mesh adaptation algorithm context where the simulation time interval [0,T] is split into n_{adap} *sub-intervals* $[t_{i-1}, t_i]$ for $i = 1, .., n_{adap}$. Each spatial mesh $\mathbf{M}^i$ is then kept constant during each sub-interval $[t_{i-1}, t_i]$. We could consider this partition as a *time discretization of*

the mesh adaptation problem. Following the previous section analysis, we deduce the optimal continuous mesh $\mathbf{M}_{L^p} = \{\mathbf{M}^i_{L^p}\}_{i=1,..,n_{adap}}$ and error:

$$\mathcal{M}^i_{L^p}(\mathbf{x}) = N_{st}^{\frac{2}{3}} \Big(\sum_{j=1}^{n_{adap}} \mathcal{K}^j \Big(\int_{t_{j-1}}^{t_j} \tau(t)^{-1} \mathrm{d}t \Big)^{\frac{2p}{2p+3}} \Big)^{-\frac{2}{3}} \Big(\int_{t_{i-1}}^{t_i} \tau(t)^{-1} \mathrm{d}t \Big)^{-\frac{2}{2p+3}} (\det \mathbf{H}^i_u(\mathbf{x}))^{-\frac{1}{2p+3}} \ \mathbf{H}^i_u(\mathbf{x}) \tag{27}$$

$$\mathbf{E}_{L^p}(\mathbf{M}_{L^p}) = 3\, N_{st}^{-\frac{2}{3}} \Big(\sum_{i=1}^{n_{adap}} \mathcal{K}^i \Big(\int_{t_{i-1}}^{t_i} \tau(t)^{-1} \mathrm{d}t \Big)^{\frac{2p}{2p+3}} \Big)^{\frac{2p+3}{3p}} , \tag{28}$$

where $\mathcal{K}^i = \left(\int_\Omega \left(\det \mathbf{H}^i_u(\mathbf{x}) \right)^{\frac{p}{2p+3}} \mathrm{d}\mathbf{x} \right)$ and matrix $\mathbf{H}^i_u$ on sub-interval i can be defined by either using an L^1 or an L^∞ norm:

$$\mathbf{H}^i_{L^1}(\mathbf{x}) = \int_{t_{i-1}}^{t_i} |H_u(\mathbf{x},t)| \, \mathrm{d}t \quad \text{or} \quad \mathbf{H}^i_{L^\infty}(\mathbf{x}) = \Delta t_i \max_{t \in [t_{i-1}, t_i]} |H_u(\mathbf{x},t)| \quad \text{with } \Delta t_i = t_i - t_{i-1} \, . \tag{29}$$

5.3.3. Mesh adaptation loop for unsteady flows

The global fixed-point mesh adaptation algorithm is schematized in Algorithm 4 where $\mathcal{H}$, $\mathcal{S}$ and $\mathcal{M}$ denote respectively meshes, solutions and metrics. And, $\mathbf{H}$ is the Hessian-metric given by Relation (29).

Algorithm 4 Mesh Adaptation Loop for Unsteady Flows

Initial mesh and solution$(\mathcal{H}_0, \mathcal{S}^0_0)$ and set targeted space-time complexity N_{st}

 # Fixed-point loop to converge the global space-time mesh adaptation problem
For $j = 1, n_{ptfx}$

 # Adaptive loop to advance the solution in time on time frame $[0, T]$

1. For $i = 1, n_{adap}$
   (a) $\mathcal{S}^j_{0,i}$ = Interpolate conservatively next sub-interval initial solution from $(\mathcal{H}^j_{i-1}, \mathcal{S}^j_{i-1}, \mathcal{H}^j_i)$;
   (b) $\mathcal{S}^j_i$ = Compute solution on sub-interval from pair $(\mathcal{S}^j_{0,i}, \mathcal{H}^j_i)$;
   (c) $|\mathbf{H}|^j_i$ = Compute sub-interval Hessian-metric from solution sample $(\mathcal{H}^j_i, \{\mathcal{S}^j_i(k)\}_{k=1,nk})$;
   EndFor
2. $\mathcal{C}^j$ = Compute space-time complexity from all Hessian-metrics $(\{|\mathbf{H}|^j_i\}_{i=1,n_{adap}})$;
3. $\{\mathcal{M}^j_i\}_{i=1,n_{adap}}$ = Compute all sub-interval unsteady metrics $(\mathcal{C}^j, \{|\mathbf{H}|^j_i\}_{i=1,n_{adap}})$;
4. $\{\mathcal{M}^j_i\}_{i=1,n_{adap}}$ = Metric gradation on all sub-interval unsteady metrics $\{\mathcal{M}^j_i\}_{i=1,n_{adap}}$;
5. $\{\mathcal{H}^{j+1}_i\}_{i=1,n_{adap}}$ = Generate all sub-interval adapted meshes $(\{\mathcal{H}^j_i, \ \mathcal{M}^j_i\}_{i=1,n_{adap}})$;

EndFor

5.3.4. A numerical example

Choice of the optimal continuous mesh. For the numerical results presented below, the following particular choice has been made to compute the optimal mesh given by Relation (27):

- the Hessian-metric for sub-interval i is discretized in time in L^1 norm: $\mathbf{H}^i_{L^1}$
- all sub-intervals have the same time length $\Delta t = T/n_{adap}$.

Practically, it remains to know how to compute the Hessian-metric $\mathbf{H}^i_{L^1}$ on sub-interval i given by Relations (29), *i.e.*, how it is discretized. The strategy adopted in [48] is to sample the solution on the time sub-interval. More precisely, n_k solutions equally distributed on the sub-interval time frame are saved, including the initial solution at t_{i-1}

and the final solution at t_i. Positive Hessian $|H_u(\mathbf{x}, t_k)|$ is evaluated for each sample. We consider the Hessian-metric $\mathbf{H}^i_{L^1}$ which is evaluated as follows:

$$\mathbf{H}^i_{L^1}(\mathbf{x}) \approx \frac{1}{2}\frac{\Delta t_i}{n_k - 1}|H_u(\mathbf{x}, t_{i-1})| + \frac{\Delta t_i}{n_k - 1}\sum_{k=2}^{n_k-1}|H_u(\mathbf{x}, t_k)| + \frac{1}{2}\frac{\Delta t_i}{n_k - 1}|H_u(\mathbf{x}, t_i)| = \Delta t_i\,|H^i_{\text{avg}}(\mathbf{x})|\,,$$

where $\Delta t_i = t_i - t_{i-1}$ is the sub-interval time length and $t_k = t_{i-1} + \frac{k-1}{n_k-1}\Delta t_i$.

Moreover, it is clear that the integral $\int_{t_{i-1}}^{t_i}\tau(t)^{-1}\mathrm{d}t$ corresponds to the number of time steps (iterations) performed by the flow solver during the i^{th} sub-interval. In practice using the flow solver number of iterations of each sub-interval to define the continuous mesh may cause trouble because it highly depends on the discrete representation of the continuous mesh, *i.e.,* the generated discrete mesh. Thus, the number of iterations of a sub-interval may substantially vary between two fixed-point iterations. To avoid this issue, one may prefer considering the flow solver time step constant on each sub-interval. In that case, we can just consider the time step $\tau(t)$ constant and equal to Δt (because of the global normalization term) so the integral $\int_{t_{i-1}}^{t_i}\tau(t)^{-1}\mathrm{d}t$ reduces to 1. Another approach is to compute the continuous time step by means of the CFL condition associated with the continuous mesh. This will cancel any issue due to the discretization. Here, we chose the first approach by considering the time step constant.

With these choices, the optimal continuous mesh $\mathbf{M}_{L^p} = \{\mathbf{M}^i_{L^p}\}_{i=1,..,n_{adap}}$ simplifies to:

$$\mathcal{M}^i_{L^p}(\mathbf{x}) = N_{st}^{\frac{2}{3}}\left(\sum_{j=1}^{n_{adap}}\Big(\int_\Omega(\det|H^j_{avg}(\mathbf{x})|)^{\frac{p}{2p+3}}\mathrm{d}\mathbf{x}\Big)\right)^{-\frac{2}{3}}\Big(\det|H^i_{avg}(\mathbf{x})|\Big)^{-\frac{1}{2p+3}}\,|H^i_{avg}(\mathbf{x})|\,.$$

A city blast. This example is a purely three-dimensional blast problem in a complex geometry representing a city. In this simulation, shock waves interact with each other and are reflected by the buildings. The city geometry is the same as in [48] with a domain size of $85\,m \times 70\,m \times 70\,m$. Initially, the ambient air is at rest $\rho_{out} = 1$ and $p_{out} = 1$. To simulate the blast, a high pressure and density region is introduced in a half-sphere of radius 2.5 m. In this region, the relevant parameters are $\rho_{in} = 10$, $p_{in} = 25$ and $\mathbf{u}_{in} = \mathbf{0}$. For both regions, we have $\gamma = 1.4$. The solution is computed until a-dimensioned time $T = 15$.

The density of the flow is chosen as the sensor variable for our mesh adaptation process. The space-time interpolation error on the sensor is controlled in L^2 norm. The time frame was split into 128 adaptation sub-intervals and 5 fixed-point iterations were used to converge the non-linear mesh adaptation problem. For each sub-interval, we consider 21 samples of the solution to build the Hessian-metric in L^1 norm. The desired accuracy was set to reach a theoretical space-time complexity N_{st} equal to $128 \times 800,000 = 102.4$ million. We start from an initial uniform mesh containing $99,255$ vertices and $549,128$ tetrahedra with a mean accuracy of 35 cm.

Figures 16 and 17 show the adapted surface and volume meshes with the corresponding iso-values of the density at the end of sub-interval 43, 86, and 128. It points out the complexity and the unpredictable behavior of the physical phenomena with a large number of shock waves interacting with the geometry. Thanks to the feature-based mesh adaptation, all shock waves are automatically captured by the adaptation process and properly refined. In the adapted meshes pictures, the mesh adaptation for a sub-interval is clearly illustrated. Additionally, the mesh refinement along band-shaped regions, which corresponds to the zone in which physical phenomena evolves during an adaptation sub-interval, are visible. At the end of the simulation, the final adapted mesh for the last sub-interval contains $4,187,548$ vertices and $25,249,618$ tetrahedra. Its maximal accuracy is about 4 mm. In Table 1, statistics of five meshes, at sub-intervals 1, 32, 64, 92, and 128, are given. We notice that the mesh accuracy is 100 times more accurate than the initial mesh allowing us to accurately capture the solution. As regards the amount of anisotropy for this simulation, an average anisotropic ratio between 12 and 18 and a mean anisotropic quotient between 106 and 270 are obtained. The anisotropic quotient measures the overall gain as compared an isotropic mesh adaptation, here more than 250 for a large part of the simulation. The gain is of course even greater when compared to a uniform mesh.

5.3.5. Chronology of the results

Figure 18 displays the evolution of the results on the city blast simulation. Left, an isotropic mesh adaptation based on a $L^\infty - L^\infty$ norm space-time error estimate done in 2003 [115]. Middle, an anisotropic mesh adaptation

based on a $L^\infty - L^p$ norm space-time error estimate done in 2011 [116]. Right, an anisotropic mesh adaptation based on the $L^p - L^p$ norm space-time error estimate presented in this article done in 2014. We clearly see the progress of the results with a lot more physics captured in the later simulations. The quality of the adapted anisotropic meshes has also been especially improved. This is due to the simultaneous advances in the error estimate, the interpolation stage, the flow solver, and the adaptive local remesher.

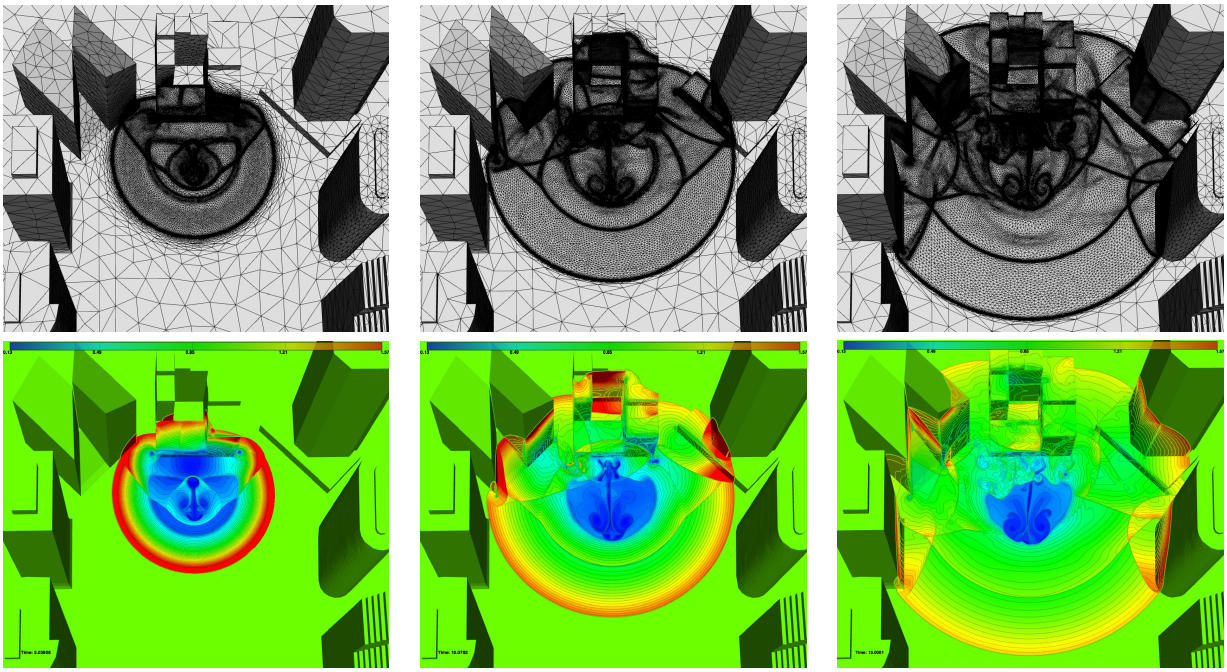


Figure 16: 3D blast in a city. Top, adapted surface meshes and, bottom, density iso-values on the surface. From left to right, meshes and solutions at sub-intervals 43, 86, and 128, respectively.

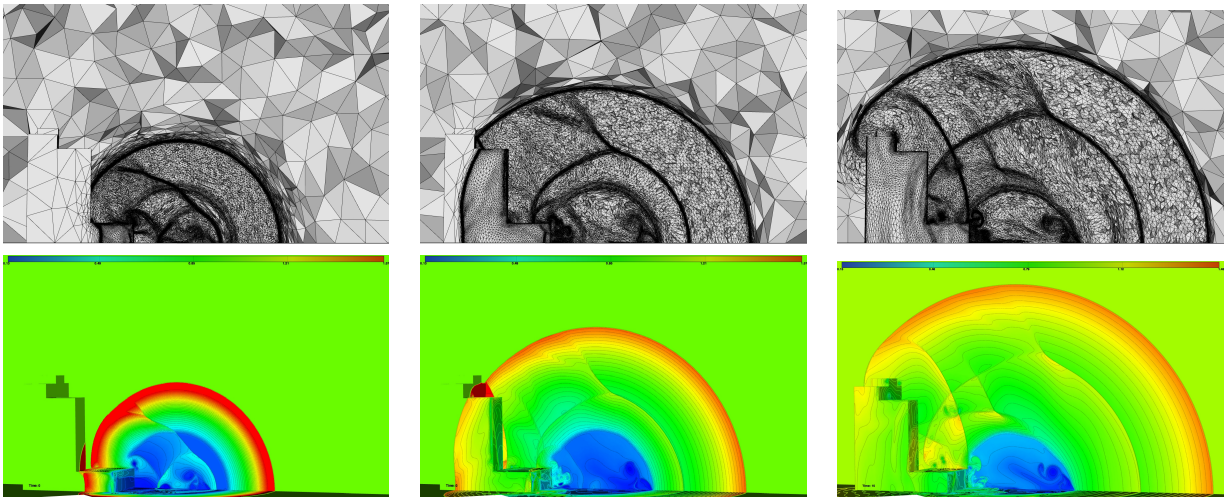
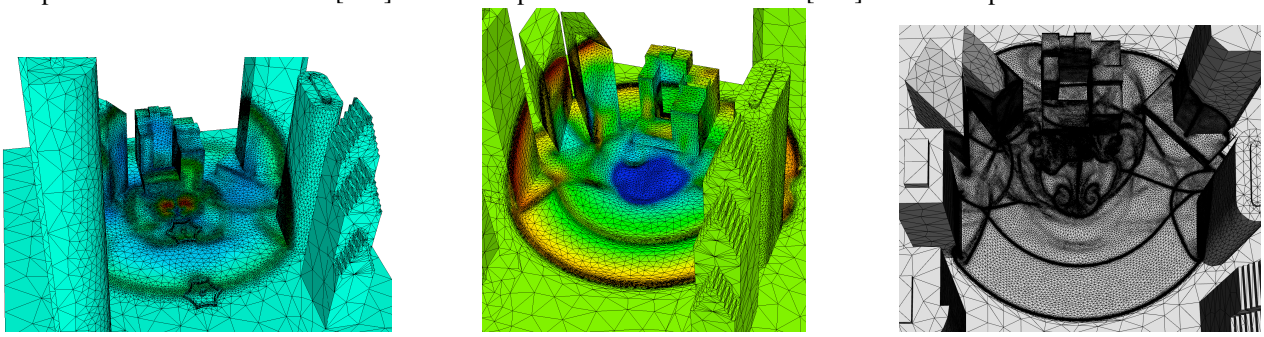


Figure 17: 3D blast in a city. Top, cut in the adapted volume meshes and, bottom, corresponding density iso-values. From left to right, meshes and solutions at sub-intervals 43, 86, and 128, respectively.

Iteration	# vertices	# tetrahedra	# triangles	min h	ratio	quotient
1	829, 275	4, 927, 323	44, 902	4 mm	12	106
32	3, 218, 351	19, 451, 629	114, 206	2 mm	18	253
64	3, 459, 242	20, 822, 072	194, 166	5 mm	18	270
92	3, 725, 108	22, 386, 629	254, 910	3.5 mm	16	251
128	4, 187, 548	25, 249, 618	329, 610	4 mm	15	248

Table 1: Mesh statistics for the 3D city blast problem.

Isotropic $L^\infty - L^\infty$ error estimate [115] | Anisotropic $L^\infty - L^p$ error estimate [116] | Anisotropic $L^p - L^p$ error estimate



743, 735 vertices
4, 328, 741 tetrahedra
87, 322 triangles
$n_{adap} = 30$ & $h_{min} > 30$ cm

185, 148 vertices
1, 027, 537 tetrahedra
50, 250 triangles
$n_{adap} = 40$ & $h_{min} = 11$ cm

4, 187, 548 vertices
25, 249, 618 tetrahedra
329, 610 triangles
$n_{adap} = 128$ & $h_{min} = 5$ mm

Figure 18: 3D blast in a city. Progress on the city blast simulation.

5.4. Goal-oriented unsteady anisotropic mesh adaptation

In the previous section, the extension of the feature-based anisotropic mesh adaptation to time-dependent problems has been detailed. It requires: a space-time L^p error analysis and a global fixed-point algorithm to converge the mesh adaptation. Now, the goal is to extend the goal-oriented mesh adaptation to transient simulations by combining the anisotropic goal-oriented error estimate and the fixed-point advances for time-accurate mesh adaptation.

To this end, several methodological issues need to be addressed. First, we propose a global fixed-point algorithm for solving the coupled system made, this time of three fields, the unsteady state, the unsteady adjoint state, and the adapted meshes. Second, this algorithm needs to be *a priori* analyzed and its convergence rate to a continuous solution needs to be optimized. Third, at the computer algorithmic level, it is also necessary to master the computational (memory and time) cost of the new system, which couples a time-forward state, a time-backward adjoint, and a mesh update influenced by global statistics.

5.4.1. Discrete adjoint system

The continuous and discrete state systems and the continuous and discrete adjoint systems are thoroughly detailed in [117]. For conciseness, we just recall the semi-discrete models. Consider the following semi-discrete unsteady compressible Euler model (explicit RK1 time integration):

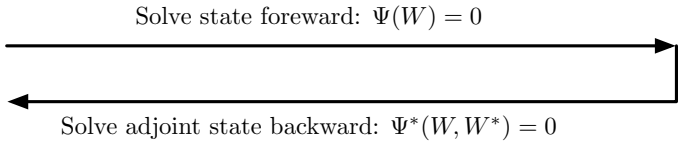
$$\Psi_h^n(W_h^n, W_h^{n-1}) \;=\; \frac{W_h^n - W_h^{n-1}}{\delta t^n} + \Phi_h(W_h^{n-1}) \;=\; 0 \quad \text{for } n = 1, ..., N.$$

The problem of minimizing the error committed on the target functional $j(W_h) = (g, W_h)$, subject to the previous Euler system, can be transformed into an unconstrained optimization problem for a given Lagrangian functional [118]. Solving this optimality problem leads to the semi-discrete unsteady adjoint model:

$$\Psi_h^{*,n}(W_h^{*,n}, W_h^{*,n-1}, W_h^{n-1}) \;=\; \frac{W_h^{*,n-1} - W_h^{*,n}}{-\delta t^n} + \Phi_h^*(W_h^{*,n}, W_h^{n-1}) \;=\; 0 \quad \text{for } n = 1, ..., N$$

$$\text{with } \Phi_h^*(W_h^{*,n}, W_h^{n-1}) \;=\; \frac{\partial j_h^{n-1}}{\partial W_h^{n-1}}(W_h^{n-1}) - (W_h^{*,n})^T \frac{\partial \Phi_h}{\partial W_h^{n-1}}(W_h^{n-1}).$$

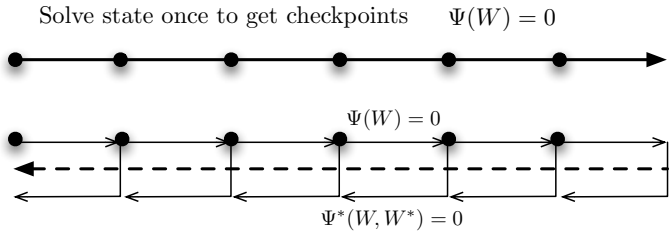
The adjoint system runs in reverse time: computing $W_h^{*,n-1}$ at time t^{n-1} requires the knowledge of state W_h^{n-1} and adjoint state $W_h^{*,n}$.



Therefore, the knowledge of all states $\left\{W_h^{n-1}\right\}_{n=1,N}$ is needed to compute backwards the adjoint state from time T to 0 which involves large memory storage effort. For instance, if we consider a 3D simulation with a mesh composed of one million vertices then we need to store at each iteration five million solution data (we have five conservative variables). If we perform $1,000$ iterations, then the memory effort to store all states is 37.25 Gb for double-type data storage (or 18.62 Gb for float-type data storage). Two strategies are employed to reduce this drawback: checkpoints and interpolation.

The memory effort can be reduced by out-of-core storage of checkpoints as shown in the picture below. First the state-simulation is performed to store checkpoints. Second, when computing backwards the adjoint, we first recompute all states from the checkpoint and store them in memory, and then we compute the unsteady adjoint until the checkpoint physical time. This method implies a recomputing effort of the state W.

The other strategy consists of storing solution states in memory for each m solver iterations only. When the unsteady adjoint is solved, solution states between two savings are linearly interpolated. This method leads to a loss of accuracy for the unsteady adjoint computation.



Solve state and backward adjoint state from checkpoints

5.4.2. Optimal unsteady adjoint-based metric

We follow the analysis of the previous sections, by neglecting boundary condition contributions and working in the continuous mesh framework, the error model for the approximation error on the functional becomes:

$$\begin{aligned} |j(W) - j(W_h)| \;&\approx\; \int_0^T \int_\Omega |W^*| \, |(W - \pi_{\mathcal{M}} W)_t| \, \mathrm{d}\Omega \, \mathrm{d}t \;+\; \int_0^T \int_\Omega |\nabla W^*| \, |\mathcal{F}(W) - \pi_{\mathcal{M}} \mathcal{F}(W)| \, \mathrm{d}\Omega \, \mathrm{d}t \\ &=\; \int_0^T \int_\Omega \mathrm{trace}\left(\mathcal{M}^{-\frac{1}{2}}(\mathbf{x},t) \, H_{go}(\mathbf{x},t) \, \mathcal{M}^{-\frac{1}{2}}(\mathbf{x},t) \right) \mathrm{d}\Omega \, \mathrm{d}t = \mathbf{E}_{go}(\mathbf{M}) \end{aligned} \tag{30}$$

$$\text{where} \qquad H_{go}(\mathbf{x},t) = \sum_{j=1}^{5} \Big([\Delta t]_j(\mathbf{x},t) + [\Delta x]_j(\mathbf{x},t) + [\Delta y]_j(\mathbf{x},t) + [\Delta z]_j(\mathbf{x},t) \Big) ,$$

$$\text{with} \quad [\Delta t]_j(\mathbf{x},t) = \left| \frac{\partial W_j^*}{\partial t}(\mathbf{x},t) \right| \cdot \left| H(W_j)(\mathbf{x},t) \right|, \quad [\Delta x]_j(\mathbf{x},t) = \left| \frac{\partial W_j^*}{\partial x}(\mathbf{x},t) \right| \cdot \left| H(\mathcal{F}_1(W_j))(\mathbf{x},t) \right|,$$

$$[\Delta y]_j(\mathbf{x},t) = \left| \frac{\partial W_j^*}{\partial y}(\mathbf{x},t) \right| \cdot \left| H(\mathcal{F}_2(W_j))(\mathbf{x},t) \right|, \quad [\Delta z]_j(\mathbf{x},t) = \left| \frac{\partial W_j^*}{\partial z}(\mathbf{x},t) \right| \cdot \left| H(\mathcal{F}_3(W_j))(\mathbf{x},t) \right|.$$

The optimal unsteady goal-oriented continuous mesh is obtained by minimizing the space-time error model:

$$\text{Find } \mathbf{M}_{go} = (\mathcal{M}_{go}(\mathbf{x},t))_{(\mathbf{x},t)\in(\Omega\times[0,T])} \text{ such that } \mathbf{E}_{go}(\mathbf{M}_{go}) = \min_{\mathbf{M}} \mathbf{E}_{go}(\mathbf{M}) \tag{31}$$

under the space-time constraint:

$$C_{st}(\mathbf{M}) = \int_0^T \tau(t)^{-1} \left(\int_\Omega d_{\mathcal{M}}(\mathbf{x},t)\, \mathrm{d}\mathbf{x} \right) \mathrm{d}t = N_{st}\,. \tag{32}$$

where $\tau(t)$ is the time step used at time t of interval $[0,T]$. Similarly to Section 5.3, after a spatial and a temporal minimization, we get the optimal continuous mesh given by Relations (27) and (29) where H_u is substituted by H_{go} and the norm p is set to 1.

5.4.3. Global adjoint fixed-point mesh adaptation algorithm

To converge the non-linear mesh adaptation problem and to compute the unsteady goal-oriented continuous meshes, the global fixed-point algorithm of Section 5.3.3 is generalized in Algorithm 5. Let us describe this algorithm. It consists of splitting the time interval $[0,T]$ into the n_{adap} mesh-adaptation time sub-intervals: $\{[t_{i-1},t_i]\}_{i=1,..,n_{adap}}$ with $t_0 = 0$ and $t_{n_{adap}} = T$. On each sub-interval a different mesh is used. A time-forward computation of the state solution is first performed with an out-of-core storage of all checkpoints, which are taken to be $\{\mathcal{W}_h(t_i)\}_{i=1,..,n_{adap}}$. Between each sub-interval, the solution is interpolated on the new mesh using the conservative interpolation. Then, starting from the last sub-interval and proceeding until the first one, we recompute and store in memory all solution states of the sub-interval from the previously stored checkpoint in order to evaluate time-backward the adjoint state throughout the sub-interval. At the same time, we evaluate the Hessian-metrics $\mathbf{H}^i_{go}$ required to generate new adapted meshes for each sub-interval. To this end, n_k Hessian-metric sample are computed on each sub-interval. At the end of the computation, the global space-time mesh complexity is evaluated, producing weights for the goal-oriented metric fields for each sub-interval. Finally, all new adapted meshes are generated according to the prescribed metrics. The time-forward, time-backward, and mesh update steps are repeated into the $j = 1,..,n_{ptfx}$ global fixed-point loop. Convergence of the fixed-point is obtained in typically five global iterations.

5.4.4. A city blast

We consider again the city blast wave propagation problem. Cost function j is the quadratic deviation from ambient pressure on target surface Γ which is composed of two buildings, see Figure 19:

$$j(W) = \int_0^T \int_\Gamma \frac{1}{2}(p(\mathbf{x},t) - p_{air})^2 \mathrm{d}\Gamma \mathrm{d}t.$$

The simulation time frame is split into 40 time sub-intervals, *i.e.,* 40 different adapted meshes are used to perform the simulation. 6 fixed-point iterations are performed to converge the mesh adaptation problem. For each sub-interval, 16 samples of the solution and adjoint states are considered to build the goal-oriented metric. We consider a space-time complexity equal to 1.2 million.

The resulting adjoint-based anisotropic adapted meshes (surface and volume) at sub-interval 10, 15, and 20 are shown in Figure 20. It is very interesting to see that we are not restricted to just one target surface. Despite the complexity and the unpredictable behavior of the physical phenomena with a large number of shock waves interactions with the geometry, the goal-oriented fixed-point mesh adaptation algorithm was automatically able to capture all shock waves which impact the targeted buildings and to set appropriate weights for the refinement of these physical phenomena. Other waves are neglected thus leading to a drastic reduction of the mesh size.

To illustrate this point, we provide meshes size for sub-intervals 1, 5, 10, and 20 in Table 2, respectively. If, for sub-interval 1, one million vertices are needed, only 40,000 vertices are used for sub-interval 20. The required mesh size has been reduced by a factor of 25 between the 1st and the 20th sub-interval. On average, almost 200,000 vertices are required to perform both adaptive computations with a maximal accuracy between 1 and 5 cm. These numbers have to be compared with uniform meshes characteristics given in Table 3 where dozens of millions of vertices are required to reach an accuracy of 30 cm.

Algorithm 5 Goal-oriented Mesh Adaptation Loop for Unsteady Flows

Initial mesh and solution($\mathcal{H}_0, \mathcal{S}_0^0$) and set targeted space-time complexity N_{st}

// Fixed-point loop to converge the global space-time mesh adaptation problem
For $j = 1, n_{ptfx}$

1. *// Adaptive loop solve state once to get checkpoints*
   For $i = 1, n_{adap}$
   (a) $\mathcal{W}_{0,i}^j$ = Interpolate conservatively next sub-interval initial sol. from $(\mathcal{H}_{i-1}^j, \mathcal{W}_{i-1}^j, \mathcal{H}_i^j)$;
   (b) $\mathcal{W}_i^j$ = Compute solution forward on sub-interval from pair $(\mathcal{W}_{0,i}^j, \mathcal{H}_i^j)$;
   EndFor
2. *// Backward adaptive loop to solve state and adjoint, and to compute Hessian-metric*
   For $i = n_{adap}, 1$
   (a) $(\mathcal{W}^*)_i^j$ = Interpolate previous sub-interval final adjoint from $(\mathcal{H}_{i+1}^j, (\mathcal{W}_0^*)_{i+1}^j, \mathcal{H}_i^j)$;
   (b) $\{\mathcal{W}_i^j(k), (\mathcal{W}^*)_i^j(k)\}_{k=1,n_k}$ = Compute state forward and adjoint backward on sub-interval and collect sample from $(\mathcal{W}_{0,i}^j, (\mathcal{W}^*)_i^j, \mathcal{H}_i^j)$;
   (c) $(|\mathbf{H}_{go}|_i^j)$ = Compute sub-interval Hessian-metric from $(\mathcal{H}_i^j, \{\mathcal{W}_i^j(k), (\mathcal{W}^*)_i^j(k)\}_{k=1,n_k})$;
   EndFor
3. C^j = Compute space-time complexity from all Hessian-metric $(\{|\mathbf{H}_{go}|_i^j\}_{i=1,n_{adap}})$;
4. $\{\mathcal{M}_i^j\}_{i=1,n_{adap}}$ = Compute all sub-interval unsteady metrics $(C^j, \{|\mathbf{H}_{go}|_i^j\}_{i=1,n_{adap}})$;
5. $\{\mathcal{H}_i^{j+1}\}_{i=1,n_{adap}}$ = Generate all sub-interval adapted meshes $(\{\mathcal{H}_i^j,\ \mathcal{M}_i^j\}_{i=1,n_{adap}})$;

EndFor

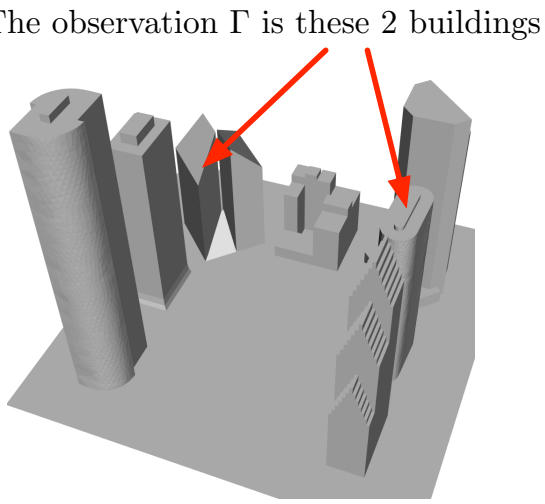


Figure 19: 3D City test case geometry and location of target surface Γ composed of two buildings.

6. Remaining challenges

A large number of open problems still remain on advanced meshing methods for scientific computing. New contributions are mandatory for mesh adaptation to handle most of the industrial applications. We briefly describe three major issues: mesh adaptation for the Navier-Stokes equations and turbulence models, mesh adaptation for 3D problems involving moving geometries, and mesh adaptation for very-high-order numerical schemes and curved meshes.

Iteration	# vertices	# tetrahedra	# triangles	min h	ratio avg. (max)	quotient avg. (max)
1	1,051,805	6,139,926	82,858	1 cm	9 (53)	67 (1,829)
10	233,116	1,327,503	36,376	1.2 cm	16 (84)	203 (4,083)
15	77,446	421,024	26,008	2.7 cm	12 (85)	124 (3,654)
20	45,500	235,383	23,072	6 cm	7 (48)	51 (1,673)
Avg.	195,355	1,113,492	31,110			

Table 2: Mesh characteristics of 3D blast wave calculation.

Uniform mesh	nv	nt	nf	average h	ratio	quotient
1	10,294,136	60,826,207	862,264	43 cm	2 (12)	3 (116)
2	33,352,859	198,163,572	2,135,032	29 cm	2 (14)	3 (165)

Table 3: Uniform mesh characteristics for the 3D city geometry.

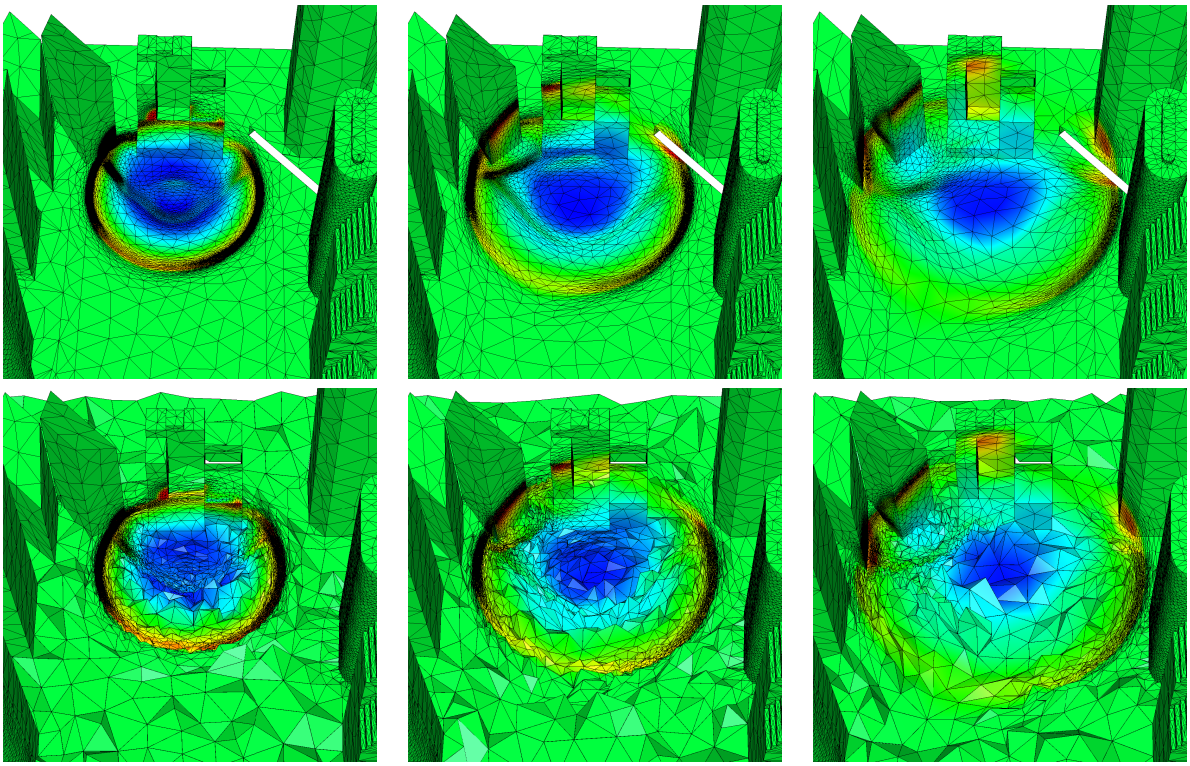


Figure 20: 3D Blast wave propagation. Adjoint-based anisotropic adapted surface (left) and volume (right) meshes at sub-interval 10, 15 and 20 and corresponding solution density at a-dimensioned time 5, 7.5, and 10.

6.1. Mesh adaptation for turbulent Navier-Stokes equations

The main conclusions of the Drag Prediction Workshop [13] and High-Lift Prediction Workshop [119] emphasized that the accuracy and convergence of high-Reynolds compressible Navier-Stokes flows rely on mesh quality. According to the results presented in this review, anisotropic mesh adaptation seems to be a good candidate to control the solution accuracy and assess the solution convergence in order to certify such numerical simulations. To this end, additional progress is needed in terms of error estimates dedicated to the Navier-Stokes equations along with specific adaptive mesh generation techniques for boundary layers.

As regards the error estimates, preliminary results regarding mesh adaptation for the laminar Navier-Stokes equa-

tions have been obtained in [120]. Other error analyses for Navier-Stokes have been introduced. In [121] an adjoint-free approach is introduced based on specific entropy variables. In [122], an extension of error estimates for elliptic problems was derived for Navier-Stokes equations. In [72, 123, 41], an approach coupling isotropic adjoint-based estimates and interpolation error is proposed. In [73], a surrogate model is used to derive anisotropic estimates. Nevertheless, turbulence models are not yet fully represented in the analysis. Future works will be directed at deriving new error estimates taking into account the considered turbulence model (RANS Spalart-Almaras, RANS $k-\varepsilon$, LES, LES-VMS, ...).

In regards to mesh adaptation techniques, a lot of work is still required to reach the same maturity as the one obtained for inviscid flows on industrial configurations. In particular, handling high Reynolds number (several millions) compressible flow simulations require suitable adaptation of the boundary layer mesh which remains a crucial open problem as it influences all the downstream flow. Indeed, most of the previous approaches keep a frozen boundary layer mesh while adapting only the far field [122, 123]. Other approaches rely on a completely unstructured boundary layer [54, 73, 94, 76]. However, in this case, additional constraints are reported on the flow solver to reduce the numerical dissipation in the boundary layer region. A few works have been conducted to adapt the boundary layer mesh. For instance, in [124], specific hybrid operators have been derived to refine the boundary layer mesh. This aspect is of crucial interest for transonic flows, where shocks may interact with the boundary layer.

Despite the great results obtained so far, adaptive meshing methods can be improved further. To efficiently solve the compressible Navier-Stokes equations, several studies point out that it is mandatory to consider structured mesh regions, in particular in the boundary layer. Recent approaches have been devoted to generate metric-aligned and metric-orthogonal meshes. It consists in taking into account the natural alignment and orthogonality of the provided input metric field during the mesh generation process. Using this supplementary information enables unstructured adaptive local remesher to automatically produce anisotropic adapted quasi-structured meshes of high-quality [97, 98, 125], see Figure 21. It is a first step in the automatic adaptation of structured boundary layer meshes.

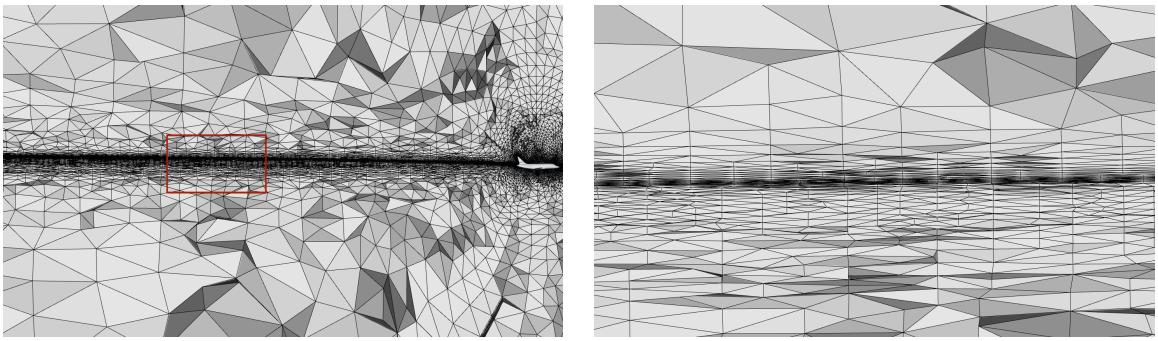


Figure 21: Examples of quasi-structured adapted meshes using the metric-orthogonal method.

6.2. *Adaptive moving mesh simulations*

More complex industrial unsteady problems involve moving geometries. These simulations are time consuming, so using mesh adaptation is a way to enhance the accuracy [126]. Immersed boundary method coupled with mesh adaptation has been proposed [127, 31, 128]. But, it is hard to extend these approaches to high Reynolds number flows because of the boundary layer. Important progress has been achieved for the robustness of body-fitted moving mesh methods by using local remeshing in order to ease the displacement of the geometry [129, 130, 58, 131]. A changing-connectivity moving mesh method has been proposed to get rid of any remeshing during the process [116, 132].

Extending mesh adaptation to this context requires to take into account the dynamic of the mesh inside the error estimate, the adapted mesh generation process, and to take into account the metric in the mesh motion. First works have been done in this direction as [116] in 2D and [133] in 3D where a specific moving mesh metric is designed, see Figure 22. Unfortunately, a lot of work remains to be done.

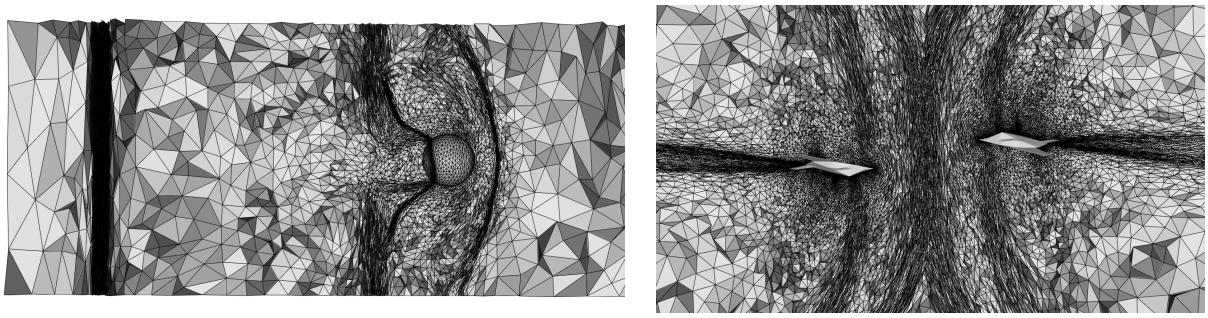


Figure 22: Examples of adapted meshes for body-fitted moving mesh simulations.

6.3. High-order adapted meshes

Numerous works focus on the design of very-high-order numerical schemes. However, all of the reviewed error estimates only consider second-order schemes. Thus, it is mandatory to extend all of our developments to higher-order estimates. The key point is to be able to efficiently reduce the multi-linear form that occurs in such a formulation into a bilinear one in order to apply the continuous mesh framework. With straight elements, a metric-based control of the interpolation error is studied in [134, 135] in 2D. In 3D, only a few works exist. For instance, Yano and Darmofal [73] introduce a Riemannian-based optimization to deduce an error estimate for Discontinuous Garlekin numerical schemes of any order on unstructured meshes. For non-confirming block-structured adapted meshes, 3D error estimates exist, see [121, 136].

The second major problematic concerns the generation of high-order curved meshes. Indeed, very-high-order numerical schemes require high-order curved meshes to represent domain boundaries in order to preserve their very-high order of accuracy. Without adaptivity, the generation of high-order curved mesh is already challenging. Recently, several works have focused on the validity and the generation of 3D meshes to comply to complex curved boundaries [137, 138, 139, 140, 141, 142].

7. Conclusion

In order to reach an optimal level of anisotropy and thus to enjoy the full benefit of unstructured anisotropic mesh adaptation (as the optimal distribution of the degrees of freedom, the improvement of the ratio accuracy with respect to the cpu time), it is necessary to control the complex interaction between the main components involved in the adaptive loop: the flow solver, the error estimate, the solution interpolation, and the mesh generator. In this paper, we show on various steady and unsteady numerical examples how the improvement of one component implies an improvement on the whole adaptive loop. From a practical point of view, the quality of the final numerical solution is also increased. Equivalently, a weakness or a wrong choice in one component may ruin the complete gain of adaptivity. This leads to the loss of mesh convergence, the loss of anisotropy, or the inability to capture all of the scales of the phenomena. Most of the improvements reviewed in this paper concern the error estimate and the adaptive mesh generator.

To derive new error estimates, we have proposed a continuous framework to model a mesh and its elements. This continuous mesh framework enforces the duality between Riemannian metric spaces and discrete meshes. This also demonstrates that metric-based mesh adaptation is well-founded.

This continuous framework is applied to feature-based anisotropic mesh adaptation. In this case, optimal meshes minimizing the interpolation error in L^p norm have been derived to control a specific field of the flow called sensor. We demonstrate that the choice of the L^p norm is primordial as it defines the sensitivity of the error estimate to the solution scales. In regards to the mesh convergence order, the theoretical second order mesh convergence is reached regardless of the chosen norm with smooth functions. However, when dealing with sharp gradients, this convergence is asymptotic and requires a number of nodes directly correlated to the amplitude of the gradient. To simulate unsteady phenomena, we have designed a new mesh adaptation algorithm which prescribes the spatial mesh of an unsteady simulation as the optimum of an error analysis. Accounting for unsteadiness is applied in a time-implicit mesh-solution coupling which needs a non-linear iteration, the fixed point. Several important issues for the

mesh adaptive space-time computation have been addressed. Among them, the strategies for choosing the splitting in time sub-intervals and the accurate integration of time errors in the mesh adaptation process have been proposed, together with a more general formulation of the mesh optimization problem in the continuous mesh framework.

This continuous framework is also used for goal-oriented anisotropic mesh adaptation to control the approximation error on a scalar-output functional. It has been derived for steady and unsteady cases. This analysis depends on the underlying numerical schemes and on the minimization of an *a priori* error estimate that is then recast in the continuous mesh framework.

Future work aims at improving the optimality of adapted meshes for time-dependent problems. Indeed, in the presented approaches, spatial and temporal domains are treated in a decoupled manner leading to a separate adaptation of the spatial and temporal meshes. Eventually, to envision a more optimal adaptation of the space-time mesh, without resorting to very frequent remeshing, the proposed idea is to consider dynamic meshes. An adequate prescribed motion of the mesh for each sub-interval may provide a more optimal answer.

For the adaptive mesh generation process, we have proposed a robust and generic cavity-based operator. Both volume and surface meshes are adapted in a coupled way in order to guarantee that a valid mesh is always provided on output. This operator handles the generation of various kind of meshes: anisotropic, quasi-structured, and hybrid. First examples of extending the standard unit-mesh approach to metric-aligned or metric-orthogonal mesh generation have been illustrated. In addition to being unit with respect to a metric, metric-orthogonal or metric-aligned meshes are composed of elements that are aligned with the eigenvectors of the metric field. This allows the recovery of locally structured meshes while keeping the same level of anisotropy. Dihedral angles are also optimized. This approach is a first step in considering fully adaptive boundary layer mesh generation.

Bibliography

[1] P. Frey, P. George, Mesh generation. Application to finite elements, ISTE Ltd and John Wiley & Sons, 2nd edn., 2008.

[2] J. Shaw, N. Weatherill, Automatic topology generation for multiblock grids, Applied Mathematics and Computation 52 (1992) 355–388.

[3] M. Aftosmis, M. Berger, S. Murman, Applications of Space-Filling Curves to Cartesian Methods for CFD, AIAA Paper 2004-1232.

[4] T. Coupez, Génération de maillages et adaptation de maillage par optimisation locale, Revue Européenne des Éléments Finis 9 (2000) 403–423.

[5] P. George, F. Hecht, E. Saltel, Automatic mesh generator with specified boundary, Comput. Methods Appl. Mech. Engrg. 92 (1991) 269–288.

[6] R. Löhner, P. Parikh, Three-dimensional grid generation by the advancing front method, Int. J. Numer. Meth. Fluids 9 (1988) 1135–1149.

[7] D. Marcum, Unstructured grid generation using automatic point insertion and local reconnection, Revue Européenne des Éléments Finis 9 (2000) 403–423.

[8] N. Weatherill, O. Hassan, Efficient three-dimensional Delaunay triangulation with automatic point creation and imposed boundary constraints, Int. J. Numer. Meth. Engng 37 (1994) 2005–2039.

[9] M. Yerry, M. Shephard, Automatic three-dimensional mesh generation by the modified-octree technique, Int. J. Numer. Meth. Engng 20 (1984) 1965–1990.

[10] D. Marcum, Efficient generation of high-quality unstructured surface and volume grids, in: Proceedings of the 9th International Meshing Roundtable, New Orleans, LA, USA, 2000.

[11] D. Mavriplis, Adaptive Meshing Techniques for Viscous Flow Calculations on Mixed Element Unstructured Meshes, Int. J. Numer. Meth. Fluids 34 (2) (2000) 93–111.

[12] G. Puigt, M. Gazaix, M. Montagnac, M.-C. LePape, M. delaLlavePlata, C. Marmignon, J.-F. Boussuge, V. Couaillier, Development of a new hybrid compressible solver inside the CFD elsA software, in: 41st AIAA Fluid Dynamics Conference and Exhibit, AIAA-2011-3048, Hawaii, HO, USA, 2011.

[13] J. C. Vassberg, E. N. Tinoco, M. Mani, B. Rider, T. Zickuhr, W. Levy, O. P. Brodersen, B. Eisfeld, S. Crippa, R. A. Wahls, J. H. Morrison, D. J. Mavriplis, M. Murayama, Summary of the fourth AIAA CFD Drag Prediction Workshop, in: 40th AIAA Fluid Dynamics Conference and Exhibit, AIAA-2010-4547, Chicago, IL, USA, 2010.

[14] J. Peraire, M. Vahdati, K. Morgan, O. Zienkiewicz, Adaptive Remeshing for Compressible Flow Computations, J. Comput. Phys. 72 (1987) 449–466.

[15] R. Löhner, Adaptive Remeshing for Transient Problems, Comput. Methods Appl. Mech. Engrg. 75 (1-3) (1989) 195–214.

[16] V. Selmin, L. Formaggia, Simulation of hypersonic flows on unstructured grids, Int. J. Numer. Meth. Engng 34 (1992) 569–606.

[17] J. Peraire, J. Peiro, K. Morgan, Adaptive Remeshing for Three-Dimensional Compressible Flow Computations, J. Comput. Phys. 103 (1992) 269–285.

[18] O. Zienkiewicz, J. Wu, Automatic directional refinement in adaptive analysis of compressible flows, Int. J. Numer. Meth. Engng 37 (1994) 2189–2210.

[19] D. Mavriplis, Adaptive mesh generation for viscous flows using Delaunay triangulation, J. Comp. Phys. 90 (1990) 271–291.

[20] P. George, F. Hecht, M. Vallet, Creation of internal points in Voronoi's type method. Control and adaptation, Adv. Eng. Software 13 (5-6) (1991) 303–312.

[21] M. Fortin, M.-G. Vallet, J. Dompierre, Y. Bourgault, W. Habashi, Anisotropic mesh adaptation : theory, validation and applications, in: Proceedings of ECCOMAS CFD, 1996.

[22] M. Castro-Díaz, F. Hecht, B. Mohammadi, O. Pironneau, Anisotropic Unstructured Mesh Adaptation for Flow Simulations, Int. J. Numer. Meth. Fluids 25 (1997) 475–491.
[23] F. Hecht, B. Mohammadi, Mesh adaptation by metric control for multi-scale phenomena and turbulence, in: 35th AIAA Aerospace Sciences Meeting and Exhibit, AIAA-1997-0859, Reno, NV, USA, 1997.
[24] J. Dompierre, M. Vallet, M. Fortin, Y. Bourgault, W. Habashi, Anisotropic mesh adaptation: towards a solver and user independent CFD, in: AIAA 35th Aerospace Sciences Meeting and Exhibit, AIAA-1997-0861, Reno, NV, USA, 1997.
[25] G. Buscaglia, E. Dari, Anisotropic Mesh Optimization and its Application in Adaptivity, Int. J. Numer. Meth. Engng 40 (1997) 4119–4136.
[26] T. Baker, Mesh adaptation strategies for problems in fluid dynamics, Finite Elem. Anal. Des. 25 (1997) 243–273.
[27] A. Tam, D. Ait-Ali-Yahia, M. Robichaud, M. Moore, V. Kozel, W. Habashi, Anisotropic mesh adaptation for 3D flows on structured and unstructured grids, Comput. Methods Appl. Mech. Engrg. 189 (2000) 1205–1230.
[28] C. Pain, A. Humpleby, C. de Oliveira, A. Goddard, Tetrahedral mesh optimisation and adaptivity for steady-state and transient finite element calculations, Comput. Methods Appl. Mech. Engrg. 190 (2001) 3771–3796.
[29] C. Bottasso, Anisotropic mesh adaption by metric-driven optimization, Int. J. Numer. Meth. Engng 60 (2004) 597–639.
[30] Y. Belhamadia, A. Fortin, E. Chamberland, Three-dimensional anisotropic mesh adaptation for phase change problems, J. Comp. Phys. 201 (2004) 753–770.
[31] C. Gruau, T. Coupez, 3D tetrahedral, unstructured and anisotropic mesh generation with adaptation to natural and multidomain metric, Comput. Methods Appl. Mech. Engrg. 194 (48-49) (2005) 4951–4976.
[32] X. Li, M. Shephard, M. Beal, 3D anisotropic mesh adaptation by mesh modification, Comput. Methods Appl. Mech. Engrg. 194 (48-49) (2005) 4915–4950.
[33] F. Alauzet, X. Li, E. S. Seol, M. Shephard, Parallel anisotropic 3D mesh adaptation by mesh modification, Engineering with Computers 21 (3) (2006) 247–258.
[34] M. Picasso, An anisotropic error indicator based on Zienkiewicz-Zhu error estimator: Application to elliptic and parabolic problems, SIAM J. Sci. Comput. 24 (4) (2003) 1328–1355.
[35] L. Formaggia, S. Micheletti, S. Perotto, Anisotropic mesh adaptation in computational fluid dynamics: Application to the advection-diffusion-reaction and the Stokes problems, Appl. Numer. Math. 51 (4) (2004) 511–533.
[36] Y. Bourgault, M. Picasso, F. Alauzet, A. Loseille, On the use of anisotropic error estimators for the adaptative solution of 3-D inviscid compressible flows, Int. J. Numer. Meth. Fluids 59 (2009) 47–74.
[37] L. Formaggia, S. Perotto, New anisotropic a priori error estimates, Numer. Math. 89 (2001) 641–667.
[38] W. Huang, Metric tensors for anisotropic mesh generation, J. Comp. Phys. 204 (2) (2005) 633–665.
[39] F. Alauzet, A. Loseille, A. Dervieux, P. Frey, Multi-dimensional continuous metric for mesh adaptation, in: Proceedings of the 15th International Meshing Roundtable, Springer, 191–214, 2006.
[40] A. Loseille, F. Alauzet, Optimal 3D highly anisotropic mesh adaptation based on the continuous mesh framework, in: Proceedings of the 18th International Meshing Roundtable, Springer, 575–594, 2009.
[41] D. Venditti, D. Darmofal, Anisotropic grid adaptation for functional outputs: application to two-dimensional viscous flows, J. Comp. Phys. 187 (1) (2003) 22–46.
[42] W. Jones, E. Nielsen, M. Park, Validation of 3D Adjoint Based Error Estimation and Mesh Adaptation for Sonic Boom Reduction, in: 44th AIAA Aerospace Sciences Meeting and Exhibit, AIAA-2006-1150, Reno, NV, USA, 2006.
[43] A. Loseille, A. Dervieux, F. Alauzet, Fully anisotropic goal-oriented mesh adaptation for 3D steady Euler equations, J. Comp. Phys. 229 (2010) 2866–2897.
[44] F. Alauzet, A. Belme, A. Dervieux, Anisotropic goal-oriented mesh adaptation for time dependent problems, in: Proceedings of the 20th International Meshing Roundtable, Springer, 99–121, 2011.
[45] F. Alauzet, A. Loseille, High Order Sonic Boom Modeling by Adaptive Methods, J. Comp. Phys. 229 (2010) 561–593.
[46] G. Compère, E. Marchandise, J.-F. Remacle, Transient adaptivity applied to two-phase incompressible flows, J. Comp. Phys. 227 (2007) 1923–1942.
[47] O. Allain, D. Guégan, F. Alauzet, Studying the impact of unstructured mesh adaptation on free surface flow simulations, in: Proceedings of the ASME 28th International Conference on Ocean, Offshore and Arctic Engineering, OMAE2009-79762, Honolulu, HI, USA, 707–715, 2009.
[48] F. Alauzet, P. Frey, P. George, B. Mohammadi, 3D transient fixed point mesh adaptation for time-dependent problems: Application to CFD simulations, J. Comp. Phys. 222 (2007) 592–623.
[49] J. Bruchon, H. Digonnet, T. Coupez, Using a signed distance function for the simulation of metal forming process: formulation of the contact condition and mesh adaptation. From Lagrangian approach to an Eulerian approach, Int. J. Numer. Meth. Engng 78 (8) (2009) 980–1008.
[50] J.-F. Remacle, X. Li, M. Shephard, J. Flaherty, Anisotropic adaptive simulation of transient flows using discontinuous Galerkin methods, Int. J. Numer. Meth. Engng 62 (2005) 899–923.
[51] F. Hecht, `BAMG`: bidimensional Anisotropic Mesh Generator, Available from http://www-rocq.inria.fr/gamma/cdrom/www/bamg/eng.htm, INRIA-Rocquencourt, France, 1998.
[52] P. Laug, H. Bourochaki, `BL2D-V2`, Mailleur bidimensionnel adaptatif, RT-0275, INRIA, 2003.
[53] P. Frey, `Yams`, A fully automatic adaptive isotropic surface remeshing procedure, RT-0252, INRIA, 2001.
[54] T. Michal, J. Krakos, Anisotropic mesh adaptation through edge primitive operations, 50th AIAA Aerospace Sciences Meeting .
[55] A. Loseille, R. Löhner, Adaptive anisotropic simulations in aerodynamics, in: 48th AIAA Aerospace Sciences Meeting, AIAA Paper 2010-169, Orlando, FL, USA, 2010.
[56] P. George, `Gamanic3d`, Adaptive anisotropic tetrahedral mesh generator, Technical Note, INRIA, 2003.
[57] G. Compère, J.-F. Remacle, J. Jansson, J. Hoffman, A mesh adaptation framework for dealing with large deforming meshes, Int. J. Numer. Meth. Engng 82 (2010) 843–867.
[58] C. Dobrzynski, P. Frey, Anisotropic Delaunay Mesh Adaptation for Unsteady Simulations, in: Proceedings of the 17th International Meshing Roundtable, Springer, 177–194, 2008.

[59] R. Löhner, Three-dimensional fluid-structure interaction using a finite element solver and adaptive remeshing, Computing Systems in Engineering 1 (2-4) (1990) 257–272.
[60] P. Frey, F. Alauzet, Anisotropic mesh adaptation for CFD computations, Comput. Methods Appl. Mech. Engrg. 194 (48-49) (2005) 5068–5082.
[61] F. Alauzet, Size gradation control of anisotropic meshes, Finite Elem. Anal. Des. 46 (2010) 181–202.
[62] A. Loseille, F. Alauzet, Continuous mesh framework. Part I: well-posed continuous interpolation error, SIAM J. Numer. Anal. 49 (1) (2011) 38–60.
[63] D. Marcum, F. Alauzet, Unstructured mesh generation using advancing layers and metric-based transition, in: 21th AIAA Computational Fluid Dynamics Conference, AIAA Paper 2013-2710, San Diego, CA, USA, 2013.
[64] A. Loseille, F. Alauzet, Continuous mesh framework. Part II: validations and applications, SIAM J. Numer. Anal. 49 (1) (2011) 61–86.
[65] A. Loseille, A. Dervieux, P. Frey, F. Alauzet, Achievement of global second-order mesh convergence for discontinuous flows with adapted unstructured meshes, in: 37th AIAA Fluid Dynamics Conference, AIAA Paper 2007-4186, Miami, FL, USA, 2007.
[66] E. Schall, D. Leservoisier, A. Dervieux, B. Koobus, Mesh adaptation as a tool for certified computational aerodynamics, Int. J. Numer. Meth. Fluids 45 (2004) 179–196.
[67] E. D'Azevedo, B. Simpson, On optimal triangular meshes for minimizing the gradient error, Numer. Math. 59 (4) (1991) 321–348.
[68] W. Cao, On the Error of Linear Interpolation and the Orientation, Aspect Ratio, and Internal Angles of a Triangle, SIAM J. Numer. Anal. 43 (1) (2005) 19–40.
[69] J.-F. Lagüe, F. Hecht, Optimal mesh for P_1 interpolation in H^1 semi-norm, in: Proceedings of the 15th International Meshing Roundtable, Springer, Birmingham, AL, USA, 259–270, 2006.
[70] P. Power, C. Pain, M. Piggott, F. Fang, G. Gorman, A. Umpleby, A. Goddard, Adjoint *a posteriori* error measures for anisotropic mesh optimization, Comput. Math. Appl. 52 (2006) 1213–1242.
[71] M. Wintzer, M. Nemec, M. Aftosmis, Adjoint-Based Adaptive Mesh Refinement for Sonic Boom Prediction, in: AIAA 26th Applied Aerodynamics Conference, AIAA-2008-6593, Honolulu, HI, USA, 2008.
[72] T. Leicht, R. Hartmann, Error estimation and anisotropic mesh refinement for 3d laminar aerodynamic flow simulations, J. Comp. Phys. 229 (19) (2010) 7344–7360.
[73] M. Yano, D. Darmofal, An optimization-based framework for anisotropic simplex mesh adaptation, J. Comp. Phys. 231 (22) (2012) 7626–7649.
[74] F. Alauzet, M. Mehrenberger, P1-conservative solution interpolation on unstructured triangular meshes, Int. J. Numer. Meth. Engng 84 (13) (2010) 1552–1588.
[75] F. Alauzet, A parallel matrix-free conservative solution interpolation on unstructured tetrahedral meshes, Comput. Methods Appl. Mech. Engrg. Submitted.
[76] V. Menier, A. Loseille, F. Alauzet, CFD Validation and Adaptivity for Viscous Flow Simulations, in: 44th AIAA Fluid Dynamics Conference, AIAA Paper 2014-2925, Atlanta, GA, USA, 2014.
[77] B. Stoufflet, J. Periaux, L. Fezoui, A. Dervieux, Numerical simulation of 3-D hypersonic Euler flows around space vehicles using adapted finite element, in: AIAA 25th Aerospace Sciences Meeting, AIAA-1987-0560, Reno, NV, USA, 1987.
[78] L. Fezoui, A. Dervieux, Finite-element non oscillatory schemes for compressible flows, in: Symposium on Computational Mathematics and Applications, vol. 730, Pavia, Italy, 1989.
[79] C. Debiez, A. Dervieux, Mixed-Element-Volume MUSCL methods with weak viscosity for steady and unsteady flow calculations, Comput. & Fluids 29 (2000) 89–118.
[80] P.-H. Cournède, B. Koobus, A. Dervieux, Positivity statements for a Mixed-Element-Volume scheme on fixed and moving grids, European Journal of Computational Mechanics 15 (7-8) (2006) 767–798.
[81] P. Roe, Approximate Riemann solvers, parameter vectors, and difference schemes, J. Comp. Phys. 43 (1981) 357–372.
[82] P. Batten, N. Clarke, C. Lambert, D. Causon, On the choice of wavespeeds for the HLLC Riemann solver, SIAM J. Sci. Comput. 18 (6) (1997) 1553–1570.
[83] C. Shu, S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes, J. Comput. Phys. 77 (1988) 439–471.
[84] R. Spiteri, S. Ruuth, A new class of optimal high-order strong-stability-preserving time discretization methods, SIAM J. Numer. Anal. 40 (2) (2002) 469–491.
[85] R. Martin, H. Guillard, A second order defect correction scheme for unsteady problems, Comput. & Fluids 25 (1996) 9–27.
[86] H. Luo, J. Baum, R. Löhner, A fast, matrix-free implicit method for compressible flows on unstructured grids, J. Comp. Phys. 146 (1998) 664–690.
[87] F. Alauzet, A. Loseille, On the use of space filling curves for parallel anisotropic mesh adaptation, in: Proceedings of the 18th International Meshing Roundtable, Springer, 337–357, 2009.
[88] D. J. Mavriplis, Unstructured Mesh Generation and Adaptivity,, Tech. Rep., ICASE Report 95-26, 1995.
[89] T. Baker, Three-dimensional mesh generation by triangulation of arbitrary point sets, in: 8th AIAA Computational Fluid Dynamics Conference, AIAA Paper 1987-1124, 1987.
[90] P.-L. George, H. Borouchaki, Delaunay triangulation and meshing : application to finite elements, Hermès Science, Paris, Oxford, 1998.
[91] P. George, F. Hecht, E. Saltel, Fully automatic mesh generator for 3D domains of any shape, Impact of Computing in Science and Engineering 2 (3) (1990) 187–218.
[92] D. L. Marcum, Efficient Generation of High-Quality Unstructured Surface and Volume Grids, Engrg. Comput. 17 (2001) 211–233.
[93] P. George, F. Hecht, Nonisotropic grids, in: Handbook of Grid Generation, Edited by J.F. Thompson, B.K. Soni and N.P. Weatherill, chap. 20, CRC Press LLC, London, New York, Washington D.C., 1998.
[94] A. Loseille, R. Löhner, Boundary layer mesh generation and adaptivity, in: 49th AIAA Aerospace Sciences Meeting, AIAA Paper 2011-0894, Orlando, FL, USA, 2011.
[95] A. Loseille, R. Löhner, On 3D anisotropic local remeshing for surface, volume and boundary layers, in: Proceedings of the 18th International Meshing Roundtable, Springer, 611–630, 2009.

[96] A. Loseille, V. Menier, Serial and Parallel Mesh Modification Through a Unique Cavity-Based Primitive, in: Proceedings of the 22th International Meshing Roundtable, Springer, 541–558, 2013.
[97] A. Loseille, Metric-orthogonal anisotropic mesh generation, Proceedings of the 23th International Meshing Roundtable, Procedia Engineering 82 (2014) 403–415.
[98] A. Loseille, D. Marcum, F. Alauzet, Alignment and orthogonality in anisotropic metric-based mesh adaptation, in: 53th AIAA Aerospace Sciences Meeting, AIAA Paper 2015-0915, Orlando, FL, USA, 2015.
[99] O. Zienkiewicz, J. Zhu, The superconvergent patch recovery and a posteriori error estimates. Part 1: The recovery technique, Int. J. Numer. Meth. Engng 33 (7) (1992) 1331–1364.
[100] R. Bank, R. Smith, A posteriori error estimate based on hierarchical bases, SIAM J. Numer. Anal. 30 (1993) 921–935.
[101] W. Huang, L. Kamenski, X. Li, A new anisotropic mesh adaptation method based upon hierarchical a posteriori error estimates, J. Comp. Phys. 229 (2010) 2179–2198.
[102] P. Clément, Approximation by finite element functions using local regularization, Revue Française d'Automatique, Informatique et Recherche Opérationnelle R-2 (1975) 77–84.
[103] O. Zienkiewicz, J. Zhu, The superconvergent patch recovery and a posteriori error estimates. Part 2: Error estimates and adaptivity, Int. J. Numer. Meth. Engng 33 (7) (1992) 1365–1380.
[104] N. Héron, F. Coulouvrat, F. Dagrau, G. Rogé, Z. Johan, HISAC midterm Overview of sonic boom issues, in: Proceedings of the 19th international congress on acoustics-ICA, Madrid, Spain, 2007.
[105] F. Alauzet, Adaptation de maillage anisotrope en trois dimensions. Application aux simulations instationnaires en Mécanique des Fluides, Ph.D. thesis, Université Montpellier II, Montpellier, France, (in French), 2003.
[106] F. Alauzet, Adaptive sonic boom sensitivity analysis, in: Proc. of the ECCOMAS CFD Conference, 2006.
[107] R. Becker, R. Rannacher, A feed-back approach to error control in finite element methods: basic analysis and examples, East-West J. Numer. Math. 4 (1996) 237–264.
[108] M. Giles, E. Suli, Adjoint methods for PDEs: a posteriori error analysis and postprocessing by duality, in: Acta Numerica, Cambridge University Press, 145–236, 2002.
[109] R. Verfürth, A review of *A Posteriori* Error Estimation and Adaptative Mesh-Refinement techniques, Wiley Teubner Mathematics, New York, 1996.
[110] R. Löhner, J. Baum, Adaptive h-refinement on 3D unstructured grids for transient problems, Int. J. Numer. Meth. Fluids 14 (12) (1992) 1407–1419.
[111] R. Rausch, J. Batina, H. Yang, Spatial adaptation procedures on tetrahedral meshes for unsteady aerodynamic flow calculations, AIAA Journal 30 (1992) 1243–1251.
[112] P. de Sampaio, P. Lyra, K. Morgan, N. Weatherill, Petrov-Galerkin solutions of the incompressible Navier-Stokes equations in primitive variables with adaptive remeshing, Comput. Methods Appl. Mech. Engrg. 106 (1993) 143–178.
[113] W. Speares, M. Berzins, A 3D unstructured mesh adaptation algorithm for time-dependent shock-dominated problems, Int. J. Numer. Meth. Fluids 25 (1997) 81–104.
[114] J. Wu, J. Zhu, J. Szmelter, O. Zienkiewicz, Error estimation and adaptivity in Navier-Stokes incompressible flows, Computational Mechanics 6 (1990) 259–270.
[115] P. Frey, F. Alauzet, Anisotropic mesh adaptation for transient flows simulations, in: Proceedings of the 12th International Meshing Roundtable, Santa Fe, New Mexico, USA, 335–348, 2003.
[116] F. Alauzet, G. Olivier, Extension of Metric-Based Anisotropic Mesh Adaptation to Time-Dependent Problems Involving Moving Geometries, in: 49th AIAA Aerospace Sciences Meeting, AIAA Paper 2011-0896, Orlando, FL, USA, 2011.
[117] A. Belme, A. Dervieux, F. Alauzet, Time accurate anisotropic goal-oriented mesh adaptation for unsteady flows, J. Comp. Phys. 231 (2012) 6323–6348.
[118] M. Giles, N. Pierce, An introduction to the adjoint approach to design, Flow, Turbulence and Combustion 65 (2000) 393–415.
[119] C. Rumsey, J. Slotnick, M. Long, R. Stuever, T. Wayman, Summary of the first AIAA CFD High-Lift Prediction Workshop, Journal of Aircraft 48 (6) (2011) 2068–2079.
[120] A. Belme, Unsteady aerodynamic and adjoint method, Ph.D. thesis, Université de Nice - Sophia Antipolis, Nice, France, 2011.
[121] K. Fidkowski, P. Roe, An entropy adjoint approach to mesh refinement, SIAM J. Sci. Comput. 32 (3) (2010) 1261–1287.
[122] W. Hassan, M. Picasso, An anisotropic adaptive finite element algorithm for transonic viscous flows around a wing, Comput. & Fluids 111 (2015) 33–45.
[123] M. Park, J. Carlson, Turbulent output-based anisotropic adaptation, in: 48th AIAA Aerospace Sciences Meeting, AIAA Paper 2010-0168, Orlando, FL, USA, 2010.
[124] A. Ovcharenko, K. Chitale, O. Sahni, K. Jansen, M. Shephard, Parallel Adaptive Boundary Layer Meshing for CFD Analysis, in: Proceedings of the 21th International Meshing Roundtable, Springer, 437–455, 2012.
[125] D. Marcum, F. Alauzet, Aligned metric-based anisotropic solution adaptive mesh generation, Proceedings of the 23th International Meshing Roundtable, Procedia Engineering 82 (2014) 428–444.
[126] R. Löhner, Three-Dimensional Fluid-Structure Interaction Using a Finite Element Solver and Adaptive Remeshing, Computing Systems in Engineering 1 (2-4) (1990) 257–272.
[127] R. Abgrall, H. Beaugendre, C. Dobrzynski, An immersed boundary method using unstructured anisotropic mesh adaptation combined with level-sets and penalization techniques, J. Comp. Phys. 257 (2014) 83–101.
[128] S. Murman, M. Aftosmis, M. Berger, Simulation of 6-DOF Motion with Cartesian Method, in: 41th AIAA Aerospace Sciences Meeting and Exhibit, AIAA-2003-1246, Reno, NV, USA, 2003.
[129] T. Baker, Adaptive modification of time evolving meshes, Comput. Methods Appl. Mech. Engrg. 194 (2005) 4977–5001.
[130] G. Compere, J.-F. Remacle, J. Jansson, J. Hoffman, A Mesh Adaptation Framework for Dealing with Large Deforming Meshes, Int. J. Numer. Meth. Engng 82 (7) (2010) 843–867.
[131] O. Hassan, E. J. Probert, K. Morgan, N. P. Weatherill, Unsteady flow simulation using unstructured meshes, Comput. Methods Appl. Mech.

Engrg. 189 (2000) 1247–1275.
[132] F. Alauzet, A changing-topology moving mesh technique for large displacement, Eng. w. Comp. 30 (2) (2014) 175–200.
[133] N. Barral, F. Alauzet, A. Loseille, Metric-Based Anisotropic Mesh Adaptation for Three-Dimensional Time-Dependent Problems Involving Moving Geometries, in: 53th AIAA Aerospace Sciences Meeting, AIAA Paper 2015-2039, Orlando, FL, USA, 2015.
[134] W. Cao, An interpolation error estimate in R^2 based on the anisotropic measures of higher order derivatives, Math. Comp. .
[135] F. Hecht, R. Kuate, An approximation of anisotropic metrics from higher order interpolation error for triangular mesh adaptation, Journal of Computational and Applied Mathematics 258 (2014) 99–115.
[136] R. Hartmann, J. Held, T. Leicht, F. Prill, Discontinuous Galerkin methods for computational aerodynamics - 3D adaptive flow simulation with the DLR PADGE code, Aerospace Science and Technology 14 (2010) 512–519.
[137] R. Abgrall, C. Dobrzynski, A. A. Froehly, A method for computing curved meshes via the linear elasticity analogy, application to fluid dynamics problems, Int. J. Numer. Meth. Fluids 76 (4) (2014) 246–266, ISSN 1097-0363.
[138] P. George, H. Borouchaki, Construction of tetrahedral meshes of degree two, Int. J. Numer. Meth. Engng 90 (2012) 1156–1182.
[139] P.-O. Persson, J. Peraire, Curved Mesh Generation and Mesh Refinement using Lagrangian Solid Mechanics, in: 47th AIAA Aerospace Sciences Meeting and Exhibit, AIAA-2009-0949, Orlando, FL, USA, 2009.
[140] O. Sahni, X. Luo, K. Jansen, M. Shephard, Curved boundary layer meshing for adaptive viscous flow simulations, Finite Elem. Anal. Des. 46 (1-2) (2010) 132–139.
[141] T. Toulorge, C. Geuzaine, J.-F. Remacle, J. Lambrechts, Robust Untangling of Curvilinear Meshes, J. Comp. Phys. 254 (2013) 8–26.
[142] Z. Q. Xie, R. Sevilla, O. Hassan, K. Morgan, The Generation of Arbitrary Order Curved Meshes for 3D Finite Element Analysis, Comput. Mech. 51 (3) (2013) 361–374.