Journal of Computational Physics 231 (2012) 6323–6348

Contents lists available at SciVerse ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Time accurate anisotropic goal-oriented mesh adaptation for unsteady flows

A. Belme [b], A. Dervieux [b], F. Alauzet [a,*]

[a] INRIA, Projet Gamma, Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay Cedex, France
[b] INRIA, Projet Tropics, 2004 route des lucioles - BP 93, 06902 Sophia Antipolis Cedex, France

ARTICLE INFO

Article history:
Received 14 October 2011
Received in revised form 19 April 2012
Accepted 6 May 2012
Available online 20 June 2012

Keywords:
Unsteady compressible flow
Goal-oriented mesh adaptation
Anisotropic mesh adaptation
Adjoint
Metric

ABSTRACT

We present a new algorithm for combining an anisotropic goal-oriented error estimate with the mesh adaptation fixed point method for unsteady problems. The minimization of the error on a functional provides both the density and the anisotropy (stretching) of the optimal mesh. They are expressed in terms of state and adjoint. This method is used for specifying the mesh for a time sub-interval. A global fixed point iterates the re-evaluation of meshes and states over the whole time interval until convergence of the space–time mesh. Applications to unsteady blast-wave and acoustic-wave Euler flows are presented.

1. Introduction

Engineering problems frequently require computational fluid dynamics (CFD) solutions with functional outputs of specified accuracy. The computational resources available for these solutions are often limited and errors in solutions and outputs are difficult to control. CFD solutions may be computed with an unnecessarily large number of mesh vertices (and associated high cost) to ensure that the outputs are computed within a required accuracy.

A powerful method for increasing the accuracy and reducing the computational cost is mesh adaptation, the aim of which is to control the accuracy of the numerical solution by changing the discretization of the computational domain according to mesh size and mesh directions constraints. This technique (i) automatically captures the anisotropy of the physical phenomena, (ii) substantially reduces the number of degrees of freedom, thus impacting favorably the CPU time, and (iii) accesses high order asymptotic convergence.

The objective of this paper is to propose a time-accurate anisotropic mesh adaptation method for functional outputs.

Pioneering works have shown a fertile development of Hessian-based or metric-based methods [11,14,16,18,21,23,25,33,35] which rely on an ideal representation of the *interpolation error* and of the *mesh*. The "multiscale" version relies on the optimization of the $\mathbf{L^p}$ norm of the interpolation error [27]. It allows to take into account the discontinuities with higher-order convergence [31]. However, these methods are limited to the minimization of some interpolation errors for some solution fields, the "sensors", and do not take into account the PDE being solved. If for many applications, this simplifying standpoint is an advantage, there are also many applications where Hessian-based mesh 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

* Corresponding author.
E-mail address: Frederic.Alauzet@inria.fr (F. Alauzet).

0021-9991/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jcp.2012.05.003

solution of the PDE. Further, in many engineering applications, a specific scalar output needs to be accurately evaluated, e.g. lift, drag, heat flux. Hessian-based adaptation does not address this issue.

On the other side, *goal-oriented* mesh adaptation focuses on deriving the best mesh to observe a given output functional. Goal-oriented methods result from a series of papers dealing with *a posteriori* estimates (see e.g. [10,20,24,34,36,37]). But, extracting informations concerning mesh anisotropy from an *a posteriori* estimate is a difficult task. Starting from *a priori* estimates, Loseille et al. proposed in [30] a fully anisotropic goal-oriented mesh adaptation technique for steady problems. This latter method combines goal-oriented rationale and the application of Hessian-based analysis to truncation error.

Mesh adaptation for unsteady flows is also an active field of research and brings an attracting increase in simulation efficiency. Complexity of the algorithms is larger than for steady case: for most flows, the mesh should change during the time interval. Meshes can be moved as in [9], pattern-split [13,26], locally refined [7], or globally rebuilt as in [2,22]. Hessian-based methods are essentially applied with a non-moving mesh system. In this paper, we do *not* account for *time discretization error* but concentrate on spatial error in unsteady simulations. A mesh adaptation fixed-point method was proposed in [2]. The Hessian criteria at the different time steps of a sub-interval are synthesized into a single criterion for these steps with the metric intersection [2,22]. A mesh-PDE solver iteration is applied on time sub-intervals. Extension to $\mathbf{L}^p$ error estimator [31] requires: (i) space–time $\mathbf{L}^\infty - \mathbf{L}^p$ error analysis, (ii) a global fixed-point algorithm to converge the mesh adaptation. This extension has been proposed in [7].

In the present work, we aim at combining the fully anisotropic goal-oriented mesh adaptation method of [30] and the fixed-point advances of [7] for time-accurate mesh adaptation.

To this end, several methodological issues need to be addressed. First, similarly to [7], we propose a global fixed-point algorithm for solving the coupled system composed 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 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.

We have started this paper with a formal description of the error analysis in its most general expression, then the application to unsteady compressible Euler flows is presented. In Section 4, we introduce the optimal adjoint-based metric definition and all its relative issues, then Section 5 is dedicated to strategies for mesh convergence and algorithm optimization. In Section 6, we present our mesh adaptation algorithm. This paper ends with numerical experiments for blast wave problems and acoustic waves.

2. Formal error analysis

Let us introduce a system of PDE's in its variational formulation:

$$\text{Find } w \in \mathcal{V} \text{ such that } \forall \varphi \in \mathcal{V}, \quad (\Psi(w), \varphi) = 0, \tag{1}$$

with $\mathcal{V}$ a functional space of solutions. The associated discrete variational formulation then writes:

$$\text{Find } w_h \in \mathcal{V}_h \text{ such that } \quad \forall \varphi_h \in \mathcal{V}_h, \quad (\Psi_h(w_h), \varphi_h) = 0, \tag{2}$$

where $\mathcal{V}_h$ is a subspace of $\mathcal{V}$. For a solution w of state system (1), we define a *functional output* as:

$$j \in \mathbb{R}; \quad j = (g, w), \tag{3}$$

where (g, w) holds for the following rather general functional output formulation:

$$(g, w) = \int_0^T \int_\Omega (g_\Omega, w) \mathrm{d}\Omega \mathrm{d}t + \int_\Omega (g_T, w(T)) \mathrm{d}\Omega + \int_0^T \int_\Gamma (g_\Gamma, w) \mathrm{d}\Gamma \mathrm{d}t, \tag{4}$$

where g_Ω, g_T, and g_Γ are assumed to be regular enough functions. We introduce the *continuous adjoint* w^*, solution of the following system:

$$w^* \in \mathcal{V}, \quad \forall \psi \in \mathcal{V}, \quad \left(\frac{\partial \Psi}{\partial w}(w)\psi, w^* \right) = (g, \psi). \tag{5}$$

The objective here is to estimate the following approximation error committed on the functional:

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

where w and w_h are respectively solutions of (1) and (2). Using the fact that $\mathcal{V}_h \subset \mathcal{V}$, the following error estimates for the unknown can be written:

$$(\Psi_h(w), \varphi_h) - (\Psi_h(w_h), \varphi_h) = (\Psi_h(w), \varphi_h) - (\Psi(w), \varphi_h) = ((\Psi_h - \Psi)(w), \varphi_h). \tag{6}$$

It is then useful to choose the test function φ_h as the discrete adjoint state, $\varphi_h = w_h^*$, which is the solution of:

$$\forall \psi_h \in \mathcal{V}_h, \quad \left(\frac{\partial \Psi_h}{\partial w_h}(w_h)\psi_h, w_h^* \right) = (g, \psi_h). \tag{7}$$

We assume that w_h^* is close to the continuous adjoint state w^*. We refer to [30] in which the following *a priori* formal estimate is finally proposed:

$$\delta j \approx ((\Psi_h - \Psi)(w), w^*). \tag{8}$$

The next section is devoted to the application of Estimator (8) to the unsteady Euler model.

3. Unsteady Euler models

3.1. Continuous state system and finite volume formulation

Continuous state system. The 3D unsteady compressible Euler equations are set in the computational space–time domain $\mathcal{Q} = \Omega \times [0,T]$, where T is the (positive) maximal time and $\Omega \subset \mathbb{R}^3$ is the spatial domain. An essential ingredient of our discretization and of our analysis is the elementwise linear interpolation operator. In order to use it easily, we define our working functional space as $V = \left[H^1(\Omega) \cap \mathcal{C}(\bar{\Omega})\right]^5$, i.e. the set of measurable functions that are continuous with square integrable gradient. We formulate the Euler model in a compact variational formulation in the functional space $\mathcal{V} = H^1\{[0,T];V\}$ as follows:

$$\begin{aligned}&\text{Find } W \in \mathcal{V} \text{ such that } \forall \varphi \in \mathcal{V}, \quad (\Psi(W),\varphi) = 0 \text{ with } (\Psi(W),\varphi)\\ &\quad = \int_\Omega \varphi(0)(W_0 - W(0))\mathrm{d}\Omega + \int_0^T \int_\Omega \varphi W_t \mathrm{d}\Omega\mathrm{d}t + \int_0^T \int_\Omega \varphi \nabla\cdot\mathcal{F}(W)\mathrm{d}\Omega\mathrm{d}t - \int_0^T \int_\Gamma \varphi \hat{\mathcal{F}}(W).\mathbf{n}\mathrm{d}\Gamma\mathrm{d}t.\end{aligned} \tag{9}$$

In the above definition, W is the vector of conservative flow variables and $\mathcal{F}(W) = (\mathcal{F}_1(W), \mathcal{F}_2(W), \mathcal{F}_3(W))$ is the usual Euler flux. The column vector W and flux tensor $\mathcal{F}$ are given by

$$W = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{pmatrix}; \quad \mathcal{F}(W) = \begin{pmatrix} \rho\mathbf{u} \\ \rho\mathbf{u}u + p\mathbf{e}_x \\ \rho\mathbf{u}v + p\mathbf{e}_y \\ \rho\mathbf{u}w + p\mathbf{e}_z \\ \rho\mathbf{u}H \end{pmatrix}. \tag{10}$$

Here ρ, p, and E represent the fluid density, thermodynamic pressure, and total energy per unit mass. u, v, and w are the Cartesian components of the velocity vector $\mathbf{u}$ and H is the total enthalpy given by $H = E + \frac{p}{\rho}$.

Functions φ and W have 5 components, and therefore the product φW holds for $\sum_{k=1..5}\varphi_k W_k$. We have denoted by Γ the boundary of the computational domain Ω, $\mathbf{n}$ is the outward normal to Γ, $W(0)(\mathbf{x}) = W(\mathbf{x},t)|_{t=0}$ for any $\mathbf{x}$ in Ω, W_0 the initial condition and the boundary flux $\hat{\mathcal{F}}$ contains the different boundary conditions, which involve inflow, outflow and slip boundary conditions.

Discrete state system. As a spatially semi-discrete model, we consider the Mixed-Element-Volume formulation [15]. As in [30], we reformulate it under the form of a finite element variational formulation, this time in the unsteady context. We assume that Ω is covered by a finite-element partition in simplicial elements denoted K. The mesh, denoted by $\mathcal{H}$ is the set of the elements. Let us introduce the following approximation space:

$$V_h = \left\{\varphi_h \in V | \varphi_{h|K} \text{ is affine } \forall K \in \mathcal{H}\right\}, \quad \text{and} \quad \mathcal{V}_h = H^1\{[0,T];V_h\} \subset \mathcal{V}.$$

Let Π_h be the usual $\mathcal{P}^1$ projector:

$$\Pi_h : V \to V_h \text{ such that } \Pi_h\varphi(\mathbf{x}_i) = \varphi(\mathbf{x}_i), \quad \forall \mathbf{x}_i \text{ vertex of } \mathcal{H}.$$

We extend it to time-dependent functions:

$$\Pi_h : H^1\{[0,T];V\} \to \mathcal{V}_h \text{ such that } (\Pi_h\varphi)(t) = \Pi_h(\varphi(t)), \quad \forall t \in [0,T].$$

The weak discrete formulation writes:

$$\begin{aligned}&\text{Find } W_h \in \mathcal{V}_h \text{ such that } \forall \varphi_h \in \mathcal{V}_h, \quad (\Psi_h(W_h),\varphi_h) = 0, \text{ with : } (\Psi_h(W_h),\varphi_h)\\ &\quad = \int_\Omega \varphi_h(0)(\Pi_h W_h(0) - W_{0h})\mathrm{d}\Omega + \int_0^T \int_\Omega \varphi_h \Pi_h W_{h,t}\mathrm{d}\Omega\mathrm{d}t + \int_0^T \int_\Omega \varphi_h \nabla\cdot\mathcal{F}_h(W_h)\mathrm{d}\Omega\mathrm{d}t - \int_0^T\\ &\quad \times \int_\Gamma \varphi_h \hat{\mathcal{F}}_h(W_h).\mathbf{n}\mathrm{d}\Gamma\mathrm{d}t + \int_0^T \int_\Omega \varphi_h D_h(W_h)\mathrm{d}\Omega\mathrm{d}t,\end{aligned} \tag{11}$$

with $\mathcal{F}_h = \Pi_h\mathcal{F}$ and $\hat{\mathcal{F}}_h = \Pi_h\hat{\mathcal{F}}$. The D_h term accounts for the numerical diffusion. In short, it involves the difference between the Galerkin central-differences approximation and a second-order Godunov approximation [15]. In the present study, we only need to know that for smooth fields, the D_h term is a third order term with respect to the mesh size. For shocked fields, monotonicity limiters become first-order terms.

Practical experiments are done with the in-house CFD software `Wolf`. The numerical scheme is vertex-centered and uses a particular edge-based formulation. This formulation consists in associating with each vertex of the mesh a control volume (or Finite-Volume cell) built by the rule of medians. This flow solver uses a HLLC approximate Riemann solver to compute numerical fluxes. A high-order scheme is derived according to a MUSCL type method using downstream and upstream tetrahedra. Appropriate β-schemes are adopted for the variable extrapolation which gives us a very high-order space-accurate scheme for the linear advection on cartesian triangular meshes. This approach provides low diffusion second-order space-accurate scheme in the non-linear case. The MUSCL type method is combined with a generalization of the Superbee limiter with three entries to guarantee the TVD property of the scheme. An explicit time stepping algorithm is used by means of multi-stages, high-order strong-stability-preserving (SSP) Runge–Kutta scheme. More details can be found in [5].

3.2. Continuous adjoint system and discretization

Continuous adjoint system. We refer here to the continuous adjoint System (5) introduced previously:

$$W^* \in \mathcal{V}, \quad \forall \psi \in \mathcal{V} : \left(\frac{\partial \Psi}{\partial W}(W)\psi, W^*\right) - (g, \psi) = 0. \tag{12}$$

We recall that (g, ψ) is defined by (4). Replacing $\Psi(W)$ by its Formulation (9) and integrating by parts, we get:

$$\left(\frac{\partial \Psi}{\partial W}(W)\psi, W^*\right) = \int_\Omega (\psi(0)W^*(0) - \psi(T)W^*(T))\mathrm{d}\Omega + \int_0^T \int_\Omega \psi\left(-W_t^* - \left(\frac{\partial \mathcal{F}}{\partial W}\right)^* \nabla W^*\right)\mathrm{d}\Omega \mathrm{d}t + \int_0^T \times \int_\Gamma \psi\left[\left(\frac{\partial \mathcal{F}}{\partial W}\right)^* W^*.\mathbf{n} - \left(\frac{\partial \hat{\mathcal{F}}}{\partial W}\right)^* W^*.\mathbf{n}\right]\mathrm{d}\Gamma \mathrm{d}t. \tag{13}$$

Consequently, the continuous adjoint state W^* must be such that:

$$-W_t^* - \left(\frac{\partial \mathcal{F}}{\partial W}\right)^* \nabla W^* = g_\Omega \text{ in } \Omega \tag{14}$$

with the associated adjoint boundary conditions:

$$\left(\frac{\partial \mathcal{F}}{\partial W}\right)^* W^*.\mathbf{n} - \left(\frac{\partial \hat{\mathcal{F}}}{\partial W}\right)^* W^*.\mathbf{n} = g_\Gamma \text{ on } \Gamma$$

and the final adjoint state condition:

$$W^*(T) = g_T.$$

The adjoint Euler equations is a system of advection equations, where the temporal integration goes backwards, i.e., in the opposite direction to usual time. Thus, when solving the unsteady adjoint system, one starts at the end of the flow run and progresses back until reaching the start time.

Discrete adjoint system. Although any consistent approximation of the continuous adjoint system could be built by discretizing System (14), we choose the option to build the discrete adjoint system from the discrete state system defined by Relation (11) in order to be closer to the true error from which the continuous model is derived.

Consider the following semi-discrete unsteady compressible Euler model (explicit RK1 time integration):

$$\Psi_h^n\left(W_h^n, W_h^{n-1}\right) = \frac{W_h^n - W_h^{n-1}}{\delta t^n} + \Phi_h\left(W_h^{n-1}\right) = 0 \quad \text{for} \quad n = 1, \ldots, N. \tag{15}$$

The time-dependent functional is discretized as follows:

$$j_h(W_h) = \sum_{n=1}^{N} \delta t^n j_h^{n-1}\left(W_h^{n-1}\right).$$

For the sake of simplicity, we restrict to the case $g_T = 0$ for the functional output defined by Relation (4). The problem of minimizing the error committed on the target functional $j(W_h) = (g, W_h)$, subject to the Euler system (15), can be transformed into an unconstrained problem for the following Lagrangian functional [19]:

$$\mathcal{L}(W_h, W_h^*) = \sum_{n=1}^{N} \delta t^n j_h^{n-1}\left(W_h^{n-1}\right) - \sum_{n=1}^{N} \delta t^n \left(W_h^{*,n}\right)^T \Psi_h^n\left(W_h^n, W_h^{n-1}\right),$$

where $W_h^{*,n}$ are the N vectors of the Lagrange multipliers (which are the time-dependent adjoint states). The conditions for an extremum are:

$$\frac{\partial \mathcal{L}}{\partial W_h^{*,n}} = 0 \quad \text{and} \quad \frac{\partial \mathcal{L}}{\partial W_h^n} = 0, \quad \text{for} \quad n = 1, \ldots, N.$$

The first condition is clearly verified from Relation (15). Thus the Lagrangian multipliers $W_h^{*,n}$ must be chosen such that the second condition of extrema is verified. This provides the unsteady discrete adjoint system:

$$\begin{cases} W_h^{*,N} &= 0 \\ W_h^{*,n-1} &= W_h^{*,n} + \delta t^n \frac{\partial j_h^{n-1}}{\partial W_h^{n-1}}\left(W_h^{n-1}\right) - \delta t^n \left(W_h^{*,n}\right)^T \frac{\partial \Phi_h}{\partial W_h^{n-1}}\left(W_h^{n-1}\right), \end{cases} \tag{16}$$

or equivalently, the semi-discrete unsteady adjoint model reads:

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

with

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

As the adjoint system runs in reverse time, the first expression in the adjoint System (16) is referred to as adjoint "initialization".

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 backward 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 millions solution data (we have 5 conservative variables). If we perform 1000 iterations, then the memory effort to store all states is 37.25 Gb for double-type data storage (or 18.62 for float-type data storage). Two strategies are employed to reduce importantly 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 backward 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 in storing solution states in memory only each m solver iterations. 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.

3.3. Impact of the adjoint: numerical example

Before going more deeply into the error model, we would like to emphasize how strongly the use of an adjoint may impact the density distribution of adapted meshes. The simulation of a blast in a 2D geometry representing a city is performed, see Fig. 1. A blast-like initialization $W_{blast} = (10, 0, 0, 250)$ in ambient air $W_{air} = (1, 0, 0, 2.5)$ is considered in a small region of the computational domain. We perform a forward/backward computation on a uniform mesh of 22 574 vertices and 44 415 triangles. Output functional of interest j is the quadratic deviation from ambient pressure on target surface S which is a part of the higher building roof (Fig. 1):

$$j(W) = \int_0^T \int_S \frac{1}{2}(p(t) - p_{air})^2 \mathrm{d}S \ \mathrm{d}t.$$

Fig. 2 plots the density isolines of the flow at different times showing several shock waves traveling throughout the computational domain. Fig. 3 depicts the associated density adjoint state progressing backward in time. The same non-dimensional physical time is considered for both figures.

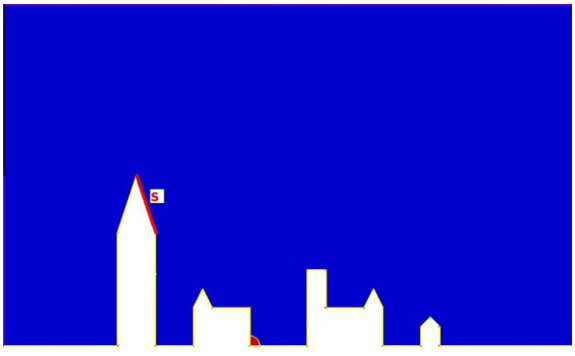

Fig. 1. Initial blast solution and location of target surface S.

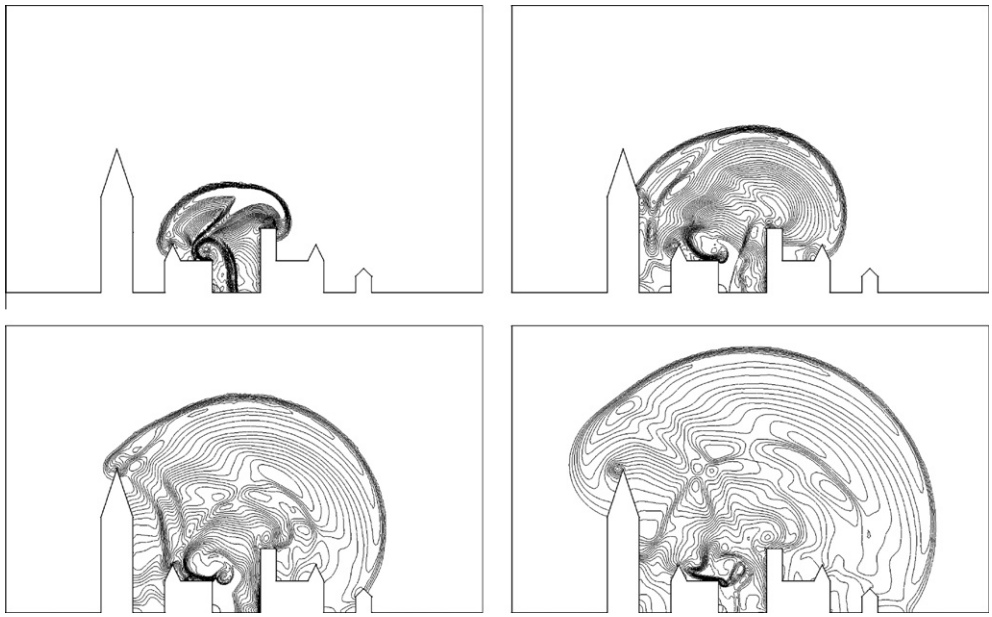

Fig. 2. 2D city blast solution state evolution. From left to right and top to bottom, snapshot of the density isolines at non-dimensional time 1.2, 2.25, 3.3 and 4.35.

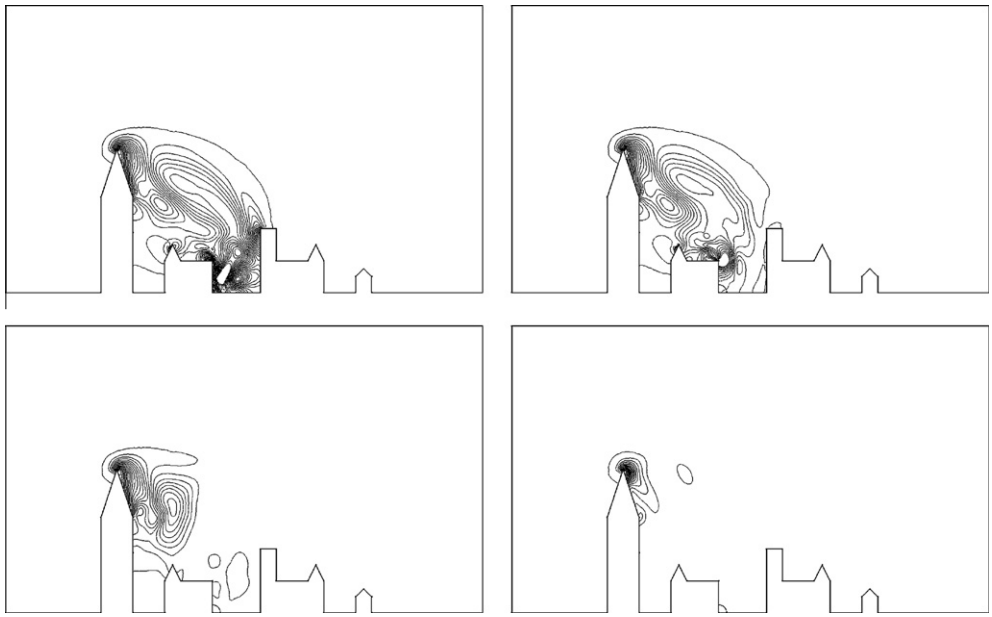

Fig. 3. 2D city blast adjoint state evolution. From left to right and top to bottom, snapshot of the adjoint-density isolines at non-dimensional time 1.2, 2.25, 3.3 and 4.35.

The simulation points out the ability of the adjoint to automatically provide the sensitivity of the flow field on the functional. Indeed, at early time of the simulation (top left picture), a lot of information is captured by the adjoint, i.e., non-zero adjoint values. We notice that shock waves which will directly impact the targeted surface are clearly detected by the adjoint, but also shocks waves reflected by the left building which will be redirected towards surface S. At the middle of the simulation, the adjoint neglects waves that are traveling in the direction opposite to S and also waves that will not impact surface S before final time T since they won't have an influence on the cost functional. While getting closer to final time T (bottom right picture), the adjoint only focuses on the last waves that will impact surface S and ignores the rest of the flow.

4. Optimal unsteady adjoint-based metric

4.1. Error analysis (applied to unsteady Euler model)

We replace in Estimation (8) operators Ψ and Ψ_h by their expressions given by Relations (9) and (11). In [30], it was observed that even for shocked flows, it is interesting to neglect the numerical viscosity term. We follow again this option. We also discard the error committed when imposing the initial condition. We derive the following simplified error model:

$$\delta j \approx \int_0^T \int_\Omega W^*(W - \Pi_h W)_t \mathrm{d}\Omega\,\mathrm{d}t + \int_0^T \int_\Omega W^* \nabla.(\mathcal{F}(W) - \Pi_h \mathcal{F}(W))\mathrm{d}\Omega\,\mathrm{d}t - \int_0^T \int_\Gamma W^*\big(\hat{\mathcal{F}}(W) - \Pi_h \hat{\mathcal{F}}(W)\big).\mathbf{n}\mathrm{d}\Gamma\,\mathrm{d}t. \tag{17}$$

Integrating by parts leads to:

$$\delta j \approx \int_0^T \int_\Omega W^*(W - \Pi_h W)_t \mathrm{d}\Omega\,\mathrm{d}t - \int_0^T \int_\Omega \nabla W^*(\mathcal{F}(W) - \Pi_h \mathcal{F}(W))\mathrm{d}\Omega\,\mathrm{d}t - \int_0^T \int_\Gamma W^*(\bar{\mathcal{F}}(W) - \Pi_h \bar{\mathcal{F}}(W)).\mathbf{n}\mathrm{d}\Gamma\,\mathrm{d}t, \tag{18}$$

with $\bar{\mathcal{F}} = \hat{\mathcal{F}} - \mathcal{F}$. We observe that this estimate of δj is expressed in terms of interpolation errors of the Euler fluxes and of the time derivative weighted by continuous functions W^* and ∇W^*.

Error bound with a safety principle. The integrands in Error Estimation (18) contain positive and negative parts which can compensate for some particular meshes. In our strategy, we prefer to not rely on these parasitic effects and to slightly overestimate the error. To this end, all integrands are bounded by their absolute values:

$$\begin{aligned}(g, W_h - W) \leqslant & \int_0^T \int_\Omega |W^*||(W - \Pi_h W)_t|\mathrm{d}\Omega\,\mathrm{d}t + \int_0^T \int_\Omega |\nabla W^*||\mathcal{F}(W) - \Pi_h \mathcal{F}(W)|\mathrm{d}\Omega\,\mathrm{d}t \\ & + \int_0^T \int_\Gamma |W^*||(\bar{\mathcal{F}}(W) - \Pi_h \bar{\mathcal{F}}(W)).\mathbf{n}|\mathrm{d}\Gamma\,\mathrm{d}t.\end{aligned} \tag{19}$$

4.2. Continuous mesh model

We propose to work in the continuous mesh framework, introduced in [28,29]. The main idea of this framework is to model discrete meshes continuously by Riemannian metric spaces. It allows us to define proper differentiable optimization [1,8], i.e., to use a calculus of variations on continuous meshes which cannot apply on the class of discrete meshes. This framework lies in the class of metric-based methods.

A continuous mesh $\mathbf{M}$ of computational domain Ω is identified to a Riemannian metric field [12] $\mathbf{M} = (\mathcal{M}(\mathbf{x}))_{\mathbf{x}\in\Omega}$. For all $\mathbf{x}$ of $\Omega, \mathcal{M}(\mathbf{x})$ is a symmetric 3×3 matrix having $(\lambda_i(\mathbf{x}))_{i=1,3}$ as eigenvalues along the principal directions $\mathcal{R}(\mathbf{x}) = (\mathbf{v}_i(\mathbf{x}))_{i=1,3}$. Sizes along these directions are denoted $(h_i(\mathbf{x}))_{i=1,3} = \left(\lambda_i^{-\frac{1}{2}}(\mathbf{x})\right)_{i=1,3}$ and the three *anisotropy quotients* r_i are defined by: $r_i = h_i^3(h_1h_2h_3)^{-1}$. The diagonalisation of $\mathcal{M}(\mathbf{x})$ writes:

$$\mathcal{M}(\mathbf{x}) = d^{\frac{2}{3}}(\mathbf{x})\mathcal{R}(\mathbf{x})\begin{pmatrix} r_1^{-\frac{2}{3}}(\mathbf{x}) & & \\ & r_2^{-\frac{2}{3}}(\mathbf{x}) & \\ & & r_3^{-\frac{2}{3}}(\mathbf{x}) \end{pmatrix}{}^t\mathcal{R}(\mathbf{x}). \tag{20}$$

The *node density* d is equal to: $d = (h_1h_2h_3)^{-1} = (\lambda_1\lambda_2\lambda_3)^{\frac{1}{2}} = \sqrt{\det(\mathcal{M})}$. By integrating the node density, we define the *complexity* $\mathcal{C}$ of a continuous mesh which is the continuous counterpart of the total number of vertices:

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

Given a continuous mesh $\mathbf{M}$, we shall say, following [28,29], that a discrete mesh $\mathcal{H}$ of the same domain Ω is a *unit mesh with respect to*$\mathbf{M}$, if each tetrahedron $K \in \mathcal{H}$, defined by its list of edges $(\mathbf{e}_i)_{i=1\ldots6}$, verifies:

$$\forall i \in [1,6], \quad \ell_{\mathcal{M}}(\mathbf{e}_i) \in \left[\frac{1}{\sqrt{2}}, \sqrt{2}\right] \quad \text{and} \quad Q_{\mathcal{M}}(K) \in [\alpha, 1] \text{ with } \alpha > 0,$$

in which the length of an edge $\ell_{\mathcal{M}}(\mathbf{e}_i)$ and the quality of an element $Q_{\mathcal{M}}(K)$ are defined as follows:

$$Q_{\mathcal{M}}(K) = \frac{36}{3^{\frac{1}{3}}}\frac{|K|_{\mathcal{M}}^{\frac{2}{3}}}{\sum_{i=1}^6 \ell_{\mathcal{M}}^2(\mathbf{e}_i)} \in [0,1], \text{ with } |K|_{\mathcal{M}} = \int_K \sqrt{\det(\mathcal{M}(\mathbf{x}))}\mathrm{d}\mathbf{x}, \text{ and } \ell_{\mathcal{M}}(\mathbf{e}_i) = \int_0^1 \sqrt{{}^t\mathbf{ab}\,\mathcal{M}(\mathbf{a}+t\mathbf{ab})\mathbf{ab}}\,\mathrm{d}t, \text{ with } \mathbf{e}_i = \mathbf{ab}.$$

We choose a tolerance α equal to 0.8.

We want to emphasize that the set of all the discrete meshes that are unit meshes with respect to a unique $\mathbf{M}$ contains an infinite number of meshes. Given a smooth function u, to each unit mesh $\mathcal{H}$ with respect to $\mathbf{M}$ corresponds a local interpolation error $|u - \Pi u|$. In [28,29], it is shown that all these interpolation errors are well represented by the so-called continuous interpolation error related to $\mathbf{M}$, which is expressed locally in terms of the Hessian H_u of u as follows:

$$(u-\pi_{\mathcal{M}}u)(\mathbf{x},t)=\frac{1}{10}\operatorname{trace}\left(\mathcal{M}^{-\frac{1}{2}}(\mathbf{x})|H_u(\mathbf{x},t)|\mathcal{M}^{-\frac{1}{2}}(\mathbf{x})\right)=\frac{1}{10}d(\mathbf{x})^{-\frac{2}{3}}\sum_{i=1}^{3}r_i(\mathbf{x})^{\frac{2}{3}t}\mathbf{v_i}(\mathbf{x})|H_u(\mathbf{x},t)|\mathbf{v_i}(\mathbf{x}), \tag{21}$$

where $|H_u|$ is deduced from H_u by taking the absolute values of its eigenvalues and where time-dependency notations have been added for use in next sections.

4.3. Continuous error model

Working in this framework enables us to write estimate (19) in a continuous form:

$$\begin{aligned}|(g,W_h-W)|\approx\mathbf{E}(\mathbf{M})=&\int_0^T\int_\Omega|W^*||(W-\pi_{\mathcal{M}}W)_t|\mathrm{d}\Omega\,\mathrm{d}t+\int_0^T\int_\Omega|\nabla W^*||\mathcal{F}(W)-\pi_{\mathcal{M}}\mathcal{F}(W)|\mathrm{d}\Omega\,\mathrm{d}t\\&+\int_0^T\int_\Gamma|W^*||(\bar{\mathcal{F}}(W)-\pi_{\mathcal{M}}\bar{\mathcal{F}}(W)).\mathbf{n}|\mathrm{d}\Gamma\,\mathrm{d}t.\end{aligned} \tag{22}$$

We observe that the third term introduces a dependency of the error on the boundary surface mesh. In the present paper, we discard this term and refer to [30] for a discussion of the influence of it. Then, introducing the continuous interpolation error, we can write the simplified error model as follows:

$$\mathbf{E}(\mathbf{M})=\int_0^T\int_\Omega\operatorname{trace}\left(\mathcal{M}^{-\frac{1}{2}}(\mathbf{x},t)\mathbf{H}(\mathbf{x},t)\mathcal{M}^{-\frac{1}{2}}(\mathbf{x},t)\right)\mathrm{d}\Omega\,\mathrm{d}t\ \text{with}\ \mathbf{H}(\mathbf{x},t)=\sum_{j=1}^{5}\left([\Delta t]_j(\mathbf{x},t)+[\Delta x]_j(\mathbf{x},t)+[\Delta y]_j(\mathbf{x},t)+[\Delta z]_j(\mathbf{x},t)\right), \tag{23}$$

in which

$$[\Delta t]_j(\mathbf{x},t)=\left|W_j^*(\mathbf{x},t)\right|\cdot|H(W_{j,t})(\mathbf{x},t)|,\quad [\Delta x]_j(\mathbf{x},t)=\left|\frac{\partial W_j^*}{\partial x}(\mathbf{x},t)\right|\cdot|H(\mathcal{F}_1(W_j))(\mathbf{x},t)|,$$
$$[\Delta y]_j(\mathbf{x},t)=\left|\frac{\partial W_j^*}{\partial y}(\mathbf{x},t)\right|\cdot|H(\mathcal{F}_2(W_j))(\mathbf{x},t)|,\quad [\Delta z]_j(\mathbf{x},t)=\left|\frac{\partial W_j^*}{\partial z}(\mathbf{x},t)\right|\cdot|H(\mathcal{F}_3(W_j))(\mathbf{x},t)|.$$

Here, W_j^* denotes the jth component of the adjoint vector W^*, $H(\mathcal{F}_i(W_j))$ the Hessian of the jth component of the vector $\mathcal{F}_i(W)$, and $H(W_{j,t})$ the Hessian of the jth component of the time derivative of W. The mesh optimization problem writes:

$$\text{Find}\ \mathbf{M}_{opt}=\operatorname{Argmin}_{\mathbf{M}}\ \mathbf{E}(\mathbf{M}), \tag{24}$$

under the constraint of bounded mesh fineness:

$$\mathcal{C}_{st}(\mathbf{M})=N_{st}, \tag{25}$$

where N_{st} is a specified total number of nodes. Since we consider an unsteady problem, the space–time (st) complexity used to compute the solution takes into account the time discretization. The above constraint then imposes the total number of nodes in the time integral, that is:

$$\mathcal{C}_{st}(\mathbf{M})=\int_0^T\tau(t)^{-1}\left(\int_\Omega d_{\mathcal{M}}(\mathbf{x},t)\mathrm{d}\mathbf{x}\right)\mathrm{d}t, \tag{26}$$

where $\tau(t)$ is the time step used at time t of interval $[0,T]$.

4.4. Spatial minimization for a fixed t

Let us assume that at time t, we seek for the optimal continuous mesh $\mathbf{M}_{go}(t)$ which minimizes the instantaneous error, i.e., the spatial error for a fixed time t:

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

under the constraint that the number of vertices is prescribed to

$$\mathcal{C}(\mathbf{M}(t))=\int_\Omega d_{\mathcal{M}(t)}(\mathbf{x},t)\mathrm{d}\mathbf{x}=N(t). \tag{27}$$

Similar to [30], solving the optimality conditions provides the *optimal goal-oriented ("go") instantaneous continuous mesh* $\mathbf{M}_{go}(t)=(\mathcal{M}_{go}(\mathbf{x},t))_{\mathbf{x}\in\Omega}$ at time t defined by:

$$\mathcal{M}_{go}(\mathbf{x},t) = N(t)^{\frac{2}{3}}\mathcal{M}_{go,1}(\mathbf{x},t), \tag{28}$$

where $\mathcal{M}_{go,1}$ is the optimum for $\mathcal{C}(\mathbf{M}(t)) = 1$:

$$\mathcal{M}_{go,1}(\mathbf{x},t) = \left(\int_\Omega (\det \mathbf{H}(\bar{\mathbf{x}},t))^{\frac{1}{5}} d\bar{\mathbf{x}}\right)^{-\frac{2}{3}} (\det \mathbf{H}(\mathbf{x},t))^{-\frac{1}{5}} \mathbf{H}(\mathbf{x},t). \tag{29}$$

The corresponding optimal instantaneous error at time t writes:

$$\tilde{\mathbf{E}}(\mathbf{M}_{go}(t)) = 3N(t)^{-\frac{2}{3}}\left(\int_\Omega (\det \mathbf{H}(\mathbf{x},t))^{\frac{1}{5}} d\mathbf{x}\right)^{\frac{5}{3}} = 3N(t)^{-\frac{2}{3}}\mathcal{K}(t). \tag{30}$$

For the sequel of this paper we denote: $\mathcal{K}(t) = \left(\int_\Omega (\det \mathbf{H}(\mathbf{x},t))^{\frac{1}{5}} d\mathbf{x}\right)^{\frac{5}{3}}$.

4.5. Temporal minimization

To complete the resolution of optimization Problem (24) and (25), we perform a temporal minimization in order to get the optimal space–time continuous mesh. In other words, we need to find the optimal time law $t \to N(t)$ for the instantaneous mesh size. First, we consider the simpler case where the time step τ is specified by the user as a function of time $t \to \tau(t)$. Second, we deal with the case of an explicit time advancing solver subject to Courant time step condition.

Temporal minimization for specified τ. Let us consider the case where the time step τ is specified by a function of time $t \to \tau(t)$. After the spatial optimization, the space–time error writes:

$$\mathbf{E}(\mathbf{M}_{go}) = \int_0^T \tilde{\mathbf{E}}(\mathbf{M}_{go}(t))dt = 3\int_0^T N(t)^{-\frac{2}{3}}\mathcal{K}(t)dt \tag{31}$$

and we aim at minimizing it under the following space–time complexity constraint:

$$\int_0^T N(t)\tau(t)^{-1}dt = N_{st}. \tag{32}$$

In other words, we concentrate on seeking for *the optimal distribution of $N(t)$ when the space–time total number of nodes N_{st} is prescribed*. Let us apply the one-to-one change of variables:

$$\tilde{N}(t) = N(t)\tau(t)^{-1} \quad \text{and} \quad \tilde{\mathcal{K}}(t) = \tau(t)^{-\frac{2}{3}}\mathcal{K}(t).$$

Then, our temporal optimization problem becomes:

$$\min_{\mathbf{M}} \mathbf{E}(\mathbf{M}) = \int_0^T \tilde{N}(t)^{-\frac{2}{3}}\tilde{\mathcal{K}}(t)dt \text{ under constraint } \int_0^T \tilde{N}(t)dt = N_{st}.$$

The solution of this problem is given by:

$$\tilde{N}_{opt}(t)^{-\frac{5}{3}}\tilde{\mathcal{K}}(t) = const \Rightarrow N_{opt}(t) = C(N_{st})(\tau(t)\mathcal{K}(t))^{\frac{3}{5}}.$$

Here, constant $C(N_{st})$ can be obtained by introducing the above expression in space–time complexity Constraint (32), leading to:

$$C(N_{st}) = \left(\int_0^T \tau(t)^{-\frac{2}{5}}\mathcal{K}(t)^{\frac{3}{5}}dt\right)^{-1} N_{st},$$

which completes the description of the optimal space–time metric for a prescribed time step. Using Relation (28), the analytic expression of the optimal space–time goal-oriented metric $\mathbf{M}_{go}$ becomes:

$$\mathcal{M}_{go}(\mathbf{x},t) = N_{st}^{\frac{2}{3}}\left(\int_0^T \tau(t)^{-\frac{2}{5}}\left(\int_\Omega (\det \mathbf{H}(\bar{\mathbf{x}},t))^{\frac{1}{5}} d\bar{\mathbf{x}}\right)dt\right)^{-\frac{2}{3}} \tau(t)^{\frac{2}{5}}(\det \mathbf{H}(\mathbf{x},t))^{-\frac{1}{5}}\mathbf{H}(\mathbf{x},t). \tag{33}$$

We get the following optimal error:

$$\mathbf{E}(\mathbf{M}_{go}) = 3N_{st}^{-\frac{2}{3}}\left(\int_0^T \tau(t)^{-\frac{2}{5}}\left(\int_\Omega (\det \mathbf{H}(\mathbf{x},t))^{\frac{1}{5}} d\mathbf{x}\right)dt\right)^{\frac{5}{3}}. \tag{34}$$

Temporal minimization for explicit time advancing. In the case of an explicit time advancing subject to a Courant condition, we get a more complex context, since time step strongly depends on the smallest mesh size. We consider only the case of smooth data and solution.

We still seek for the optimal continuous mesh that minimizes space–time Error (31) under complexity Constraint (32). Let $\Delta x_{min,1}(t) = \min_{\mathbf{x}}\min_i h_i(\mathbf{x})$ be the smallest mesh size of $\mathbf{M}_{go,1}(t)$. Since the metric Definition (20) is homogeneous with the inverse square of mesh size, we deduce the smallest mesh size of $\mathbf{M}_{go}(t)$ from Relation (28):

$$\Delta x_{min}(t) = N(t)^{-\frac{1}{3}}\Delta x_{min,1}(t),$$

where $\Delta x_{min,1}(t)$ is independent of the mesh complexity. A way to write the Courant condition for time-advancing is to define the time step $\tau(t)$ by:

$$\tau(t) = c(t)^{-1}\Delta x_{min}(t) = N(t)^{-\frac{1}{3}}c(t)^{-1}\Delta x_{min,1}(t),$$

where $c(t)$ is the maximal wave speed over the domain at time t. Again, we search for the optimal distribution of $N(t)$ when the space–time complexity N_{st} is prescribed (Relation (32)), with

$$N_{st} = \int_0^T N(t)^{\frac{4}{3}}c(t)(\Delta x_{min,1}(t))^{-1}\mathrm{d}t.$$

We choose to apply the one-to-one change of variables:

$$\hat{N}(t) = N(t)^{\frac{4}{3}}c(t)(\Delta x_{min,1}(t))^{-1} \quad \text{and} \quad \hat{\mathcal{K}}(t) = \mathcal{K}(t)c(t)^{\frac{1}{2}}(\Delta x_{min,1}(t))^{-\frac{1}{2}}.$$

Therefore, the corresponding space–time approximation error over the simulation time interval and space–time complexity reduces to:

$$\mathbf{E}(\mathbf{M}_{go}) = 3\int_0^T N(t)^{-\frac{2}{3}}\mathcal{K}(t)\mathrm{d}t = 3\int_0^T \hat{N}(t)^{-\frac{1}{2}}\hat{\mathcal{K}}(t)\mathrm{d}t \quad \text{and} \quad \int_0^T \hat{N}(t)\mathrm{d}t = N_{st}.$$

This optimization problem has for solution:

$$\hat{N}_{opt}(t)^{-\frac{3}{2}}\hat{\mathcal{K}}(t) = const \Rightarrow \hat{N}_{opt}(t) = C(N_{st})\hat{\mathcal{K}}(t)^{\frac{2}{3}}$$

and by considering the space–time complexity constraint relation we deduce:

$$C(N_{st}) = N_{st}\left(\int_0^T \hat{\mathcal{K}}(t)^{\frac{2}{3}}\mathrm{d}t\right)^{-1}.$$

Using the definitions of $\hat{N}$ and $\hat{\mathcal{K}}$ in the above relations, we get:

$$\begin{aligned} N(t)^{\frac{4}{3}}c(t)(\Delta x_{min,1}(t))^{-1} &= N_{st}\left(\int_0^T \left(\mathcal{K}(t)c(t)^{\frac{1}{2}}(\Delta x_{min,1}(t))^{-\frac{1}{2}}\right)^{\frac{2}{3}}\mathrm{d}t\right)^{-1}\left(\mathcal{K}(t)c(t)^{\frac{1}{2}}(\Delta x_{min,1}(t))^{-\frac{1}{2}}\right)^{\frac{2}{3}} \Longleftrightarrow N(t) \\ &= N_{st}^{\frac{3}{4}}c(t)^{-\frac{1}{2}}(\Delta x_{min,1}(t))^{\frac{1}{2}}\mathcal{K}(t)^{\frac{1}{2}}\left(\int_0^T c(t)^{\frac{1}{3}}(\Delta x_{min,1}(t))^{-\frac{1}{3}}\mathcal{K}(t)^{\frac{2}{3}}\mathrm{d}t\right)^{-\frac{3}{4}}. \end{aligned}$$

Consequently, after some simplifications, we obtain the following expression of the optimal space–time goal-oriented continuous mesh $\mathbf{M}_{go}$ and error:

$$\mathcal{M}_{go}(\mathbf{x},t) = N_{st}^{\frac{1}{2}}\left(\int_0^T \theta(t)^{\frac{1}{3}}\mathcal{K}(t)^{\frac{2}{3}}\mathrm{d}t\right)^{-\frac{1}{2}}\theta(t)^{-\frac{1}{3}}\mathcal{K}(t)^{-\frac{1}{15}}(\det\mathbf{H}(\mathbf{x},t))^{-\frac{1}{5}}\mathbf{H}(\mathbf{x},t) \tag{35}$$

$$\mathbf{E}(\mathbf{M}_{go}) = 3N_{st}^{-\frac{1}{2}}\left(\int_0^T \theta(t)^{\frac{1}{3}}\mathcal{K}(t)^{\frac{2}{3}}\mathrm{d}t\right)^{\frac{3}{2}}, \tag{36}$$

where $\theta(t) = c(t)(\Delta x_{min,1}(t))^{-1}$ and $\mathcal{K}(t) = \left(\int_\Omega(\det\mathbf{H}(\mathbf{x},t))^{\frac{1}{5}}\mathrm{d}\mathbf{x}\right)^{\frac{5}{3}}$.

4.6. Temporal minimization for time sub-intervals

The previous analysis provides the optimal size of the adapted meshes for each time level. Therefore, this analysis requires the mesh to be adapted at each flow solver time step. However, in practice this approach involves a very large number of remeshing which is CPU consuming and spoils solution accuracy due to the many solution transfers from one mesh to a new one. As a consequence, a new adaptive strategy has been proposed in [2,7] where the number of remeshing is controlled (thus drastically reduced) by generating adapted meshes for several solver time steps. The idea is to split the simulation time interval into n_{adap} *sub-intervals* $[t_{i-1}, t_i]$ for $i = 1, \ldots, n_{adap}$. 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*. In other words, the number of nodes N^i of the i^{th} adapted mesh $\mathbf{M}^i$ on sub-interval $[t_{i-1}, t_i]$ should for example be taken equal to:

$$N^i = \frac{\int_{t_{i-1}}^{t_i} N_{opt}(t)\tau(t)^{-1}\mathrm{d}t}{\int_{t_{i-1}}^{t_i} \tau(t)^{-1}\mathrm{d}t}.$$

Here, we propose a different option in which we get an optimal discrete answer.

Spatial minimization on a sub-interval. Given the continuous mesh complexity N^i for the single adapted mesh used during time sub-interval $[t_{i-1}, t_i]$, we seek for the optimal continuous mesh $\mathbf{M}_{go}^i$ solution of the following problem:

$$\min_{\mathbf{M}^i} \mathbf{E}^i(\mathbf{M}^i) = \int_\Omega \text{trace}\Big((\mathcal{M}^i)^{-\frac{1}{2}}(\mathbf{x})\mathbf{H}^i(\mathbf{x})(\mathcal{M}^i)^{-\frac{1}{2}}(\mathbf{x})\Big)\mathrm{d}\mathbf{x} \text{ such that } \mathcal{C}(\mathbf{M}^i) = N^i,$$

where matrix $\mathbf{H}^i$ on the sub-interval can be defined by either using an $\mathbf{L}^1$ or an $\mathbf{L}^\infty$ norm:

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

with $\Delta t_i = t_i - t_{i-1}$. Proceeding as before, we get the spatial optimality condition:

$$\mathcal{M}^i_{go}(\mathbf{x}) = (N^i)^{\frac{2}{5}}\mathcal{M}^i_{go,1}(\mathbf{x}) \text{ with } \mathcal{M}^i_{go,1}(\mathbf{x}) = \left(\int_\Omega (\det \mathbf{H}^i(\bar{\mathbf{x}}))^{\frac{1}{5}}\mathrm{d}\bar{\mathbf{x}}\right)^{-\frac{2}{3}} (\det \mathbf{H}^i(\mathbf{x}))^{-\frac{1}{5}}\mathbf{H}^i(\mathbf{x}).$$

The corresponding optimal error $\mathbf{E}^i(\mathbf{M}^i_{go})$ writes:

$$\mathbf{E}^i(\mathbf{M}^i_{go}) = 3(N^i)^{-\frac{2}{3}}\left(\int_\Omega (\det \mathbf{H}^i(\mathbf{x}))^{\frac{1}{5}}\mathrm{d}\mathbf{x}\right)^{\frac{5}{3}} = 3(N^i)^{-\frac{2}{3}}\mathcal{K}^i.$$

To complete our analysis, we shall perform a temporal minimization. Again, we first consider the case where the time step τ is specified by a function of time and, secondly, we then deal with the case of an explicit time advancing solver subject to Courant time step condition.

Temporal minimization for specified τ. After the spatial minimization, the temporal optimization problem reads:

$$\min_{\mathbf{M}} \mathbf{E}(\mathbf{M}) = \sum_{i=1}^{n_{adap}} \mathbf{E}^i(\mathbf{M}^i_{go}) = 3\sum_{i=1}^{n_{adap}} (N^i)^{-\frac{2}{3}}\mathcal{K}^i \text{ such that } \sum_{i=1}^{n_{adap}} N^i\left(\int_{t_{i-1}}^{t_i} \tau(t)^{-1}\mathrm{d}t\right) = N_{st}.$$

We set the one-to-one mapping:

$$\tilde{N}^i = N^i\left(\int_{t_{i-1}}^{t_i} \tau(t)^{-1}\mathrm{d}t\right) \quad \text{and} \quad \tilde{\mathcal{K}}^i = \mathcal{K}^i\left(\int_{t_{i-1}}^{t_i} \tau(t)^{-1}\mathrm{d}t\right)^{\frac{2}{3}},$$

then the optimization problem reduces to:

$$\min_{\mathbf{M}} \sum_{i=1}^{n_{adap}} (\tilde{N}^i)^{-\frac{2}{3}}\tilde{\mathcal{K}}^i \text{ such that } \sum_{i=1}^{n_{adap}} \tilde{N}^i = N_{st}.$$

The solution is:

$$\tilde{N}^i_{opt} = \mathcal{C}(N_{st})(\tilde{\mathcal{K}}^i)^{\frac{3}{5}} \text{ with } \mathcal{C}(N_{st}) = N_{st}\left(\sum_{i=1}^{n_{adap}} (\tilde{\mathcal{K}}^i)^{\frac{3}{5}}\right)^{-1} \Rightarrow N^i = N_{st}\left(\sum_{i=1}^{n_{adap}} (\mathcal{K}^i)^{\frac{3}{5}}\left(\int_{t_{i-1}}^{t_i} \tau(t)^{-1}\mathrm{d}t\right)^{\frac{2}{5}}\right)^{-1} (\mathcal{K}^i)^{\frac{3}{5}}\left(\int_{t_{i-1}}^{t_i} \tau(t)^{-1}\mathrm{d}t\right)^{-\frac{3}{5}}$$

and we deduce the following optimal continuous mesh $\mathbf{M}_{go} = \{\mathbf{M}^i_{go}\}_{i=1,\dots,n_{adap}}$ and error:

$$\mathcal{M}^i_{go}(\mathbf{x}) = N_{st}^{\frac{2}{3}}\left(\sum_{i=1}^{n_{adap}} (\mathcal{K}^i)^{\frac{3}{5}}\left(\int_{t_{i-1}}^{t_i} \tau(t)^{-1}\mathrm{d}t\right)^{\frac{2}{5}}\right)^{-\frac{2}{3}} \left(\int_{t_{i-1}}^{t_i} \tau(t)^{-1}\mathrm{d}t\right)^{-\frac{2}{5}} (\det \mathbf{H}^i(\mathbf{x}))^{-\frac{1}{5}}\mathbf{H}^i(\mathbf{x}) \tag{37}$$

$$\mathbf{E}(\mathbf{M}_{go}) = 3N_{st}^{-\frac{2}{3}}\left(\sum_{i=1}^{n_{adap}} (\mathcal{K}^i)^{\frac{3}{5}}\left(\int_{t_{i-1}}^{t_i} \tau(t)^{-1}\mathrm{d}t\right)^{\frac{2}{5}}\right)^{\frac{5}{3}}, \tag{38}$$

with $(\mathcal{K}^i)^{\frac{3}{5}} = \int_\Omega (\det \mathbf{H}^i(\mathbf{x}))^{\frac{1}{5}}\mathrm{d}\mathbf{x}$.

Temporal minimization for explicit time advancing. Similarly to the previous section, the Courant-based time-step writes:

$$\tau(t) = c(t)^{-1}\Delta x^i_{min} = (N^i)^{-\frac{1}{3}}c(t)^{-1}\Delta x^i_{min,1} \quad \text{for} \quad t \in [t_{i-1}, t_i],$$

where $\Delta x^i_{min,1}$ is the smallest altitude of $\mathbf{M}^i_{go,1}$ and $c(t)$ is the maximal wave speed over the domain. The optimization problem writes:

$$\min_{\mathbf{M}} \mathbf{E}(\mathbf{M}) = \sum_{i=1}^{n_{adap}} \mathbf{E}^i(\mathbf{M}^i_{go}) = 3\sum_{i=1}^{n_{adap}} (N^i)^{-\frac{2}{3}}\mathcal{K}^i$$

under the constraint:

$$\sum_{i=1}^{n_{adap}} (N^i)^{\frac{4}{3}}\left(\int_{t_{i-1}}^{t_i} c(t)(\Delta x^i_{min,1})^{-1}\mathrm{d}t\right) = N_{st}.$$

We set again:

$$\hat{N}^i = \left(N^i\right)^{\frac{4}{3}} \left(\int_{t_{i-1}}^{t_i} c(t) (\Delta x^i_{min,1})^{-1} \mathrm{d}t \right) \quad \text{and} \quad \hat{\mathcal{K}}^i = \mathcal{K}^i \left(\int_{t_{i-1}}^{t_i} c(t) (\Delta x^i_{min,1})^{-1} \mathrm{d}t \right)^{\frac{1}{2}}.$$

Then, the optimization problem becomes:

$$\min_{\mathbf{M}} \mathbf{E}(\mathbf{M}) = \sum_{i=1}^{n_{adap}} \mathbf{E}^i(\mathbf{M}^i_{go}) = 3 \sum_{i=1}^{n_{adap}} (N^i)^{-\frac{2}{3}} \mathcal{K}^i = 3 \sum_{i=1}^{n_{adap}} (\hat{N}^i)^{-\frac{1}{2}} \hat{\mathcal{K}}^i$$

under the constraint:

$$\sum_{i=1}^{n_{adap}} (N^i)^{\frac{4}{3}} \left(\int_{t_{i-1}}^{t_i} c(t) (\Delta x^i_{min,1})^{-1} \mathrm{d}t \right) = \sum_{i=1}^{n_{adap}} \hat{N}^i = N_{st}.$$

This optimization problem has for solution:

$$\hat{N}^i_{opt} = \mathcal{C}(N_{st}) (\hat{\mathcal{K}}^i)^{\frac{2}{3}} \text{ with } \mathcal{C}(N_{st}) = N_{st} \left(\sum_{i=1}^{n_{adap}} (\hat{\mathcal{K}}^i)^{\frac{2}{3}} \right)^{-1},$$

from which we deduce:

$$N^i_{opt} = N_{st}^{\frac{3}{4}} \left(\sum_{i=1}^{n_{adap}} (\mathcal{K}^i)^{\frac{2}{3}} \left(\int_{t_{i-1}}^{t_i} c(t) (\Delta x^i_{min,1})^{-1} \mathrm{d}t \right)^{\frac{1}{3}} \right)^{-\frac{3}{4}} (\mathcal{K}^i)^{\frac{1}{2}} \left(\int_{t_{i-1}}^{t_i} c(t) (\Delta x^i_{min,1})^{-1} \mathrm{d}t \right)^{-\frac{1}{2}}.$$

For the sake of clarity, we set: $\theta^i = \int_{t_{i-1}}^{t_i} c(t) (\Delta x^i_{min,1})^{-1} \mathrm{d}t$. The optimal continuous mesh $\mathbf{M}_{go} = \{\mathbf{M}^i_{go}\}_{i=1,\dots,n_{adap}}$ and error read:

$$\mathcal{M}^i_{go}(\mathbf{x}) = N_{st}^{\frac{1}{2}} \left(\sum_{i=1}^{n_{adap}} (\mathcal{K}^i)^{\frac{2}{3}} (\theta^i)^{\frac{1}{3}} \right)^{-\frac{1}{2}} (\theta^i)^{-\frac{1}{3}} (\mathcal{K}^i)^{-\frac{1}{15}} (\det \mathbf{H}^i(\mathbf{x}))^{-\frac{1}{5}} \mathbf{H}^i(\mathbf{x}) \tag{39}$$

$$\mathbf{E}(\mathbf{M}_{go}) = 3 N_{st}^{-\frac{1}{2}} \left(\sum_{i=1}^{n_{adap}} (\mathcal{K}^i)^{\frac{2}{3}} (\theta^i)^{\frac{1}{3}} \right)^{\frac{3}{2}}. \tag{40}$$

Remark 1. The choice of the time sub-intervals $\{[t_{i-1}, t_i]\}_{i=1,n_{adap}}$ for a given n_{adap} is a mesh adaptation problem: what is the subdivision of interval $[0, T]$ in n_{adap} sub-intervals which will minimize the error on optimal mesh/metric $\mathcal{M}$? Since we take a constant metric in the sub-interval, the error main term in approximating $\mathcal{M}$ is the following integral of the absolute value of the time-derivative of $\mathcal{M}$:

$$\sum_{i=1}^{n_{adap}} \int_{t_{i-1}}^{t_i} \left| \frac{\partial \mathcal{M}(t)}{\partial t} \right| \mathrm{d}t,$$

which can be minimized for instance by isodistribution of the error by sub-interval.

Remark 2. The parameter n_{adap}, number of time sub-intervals is important for the efficiency of the adaptation algorithm. When a too large value is prescribed for n_{adap}, the error in mesh-to-mesh interpolation may increase. A compromise needs to be found by the user.

5. Theoretical mesh convergence analysis

Our motivation for developing mesh adaptation algorithms is to obtain approximation algorithms with better convergence to continuous target data. By better, we mean that we want to reach the asymptotic high order convergence with a lower number of nodes and also for solutions involving discontinuities. Both improvements have been previously obtained in the context of steady inviscid flows [5]. The present paper focuses on mesh adaptation only controlling the spatial approximation error in the context of unsteady flows. From this standpoint, we just forget about time approximation error by specifying a time step or by considering an explicit time advancing context close to the one of references [2,7,22]. We assume that accuracy of the time advancing scheme is second-order at least. Then it can be derived from basic arguments that the time approximation error is essentially smaller than or equal to the spatial approximation error, controlled by the metric-based method, which justifies to adapt the mesh only to spatial error [2,7,22].

In order to measure the theoretical convergence order of the mesh adaptation algorithm, we need to compare the global mesh effort -the complexity N_{st}- with the corresponding error level. We recall that an adaptative or a uniform discretisation approach both using N degrees of freedom has a convergence of order α if the approximation error $|u - u_N|$ satisfies:

$$|u - u_N| \leqslant const.N^{-\frac{\alpha}{d}},$$

where d is the Euclidian dimension of the computational domain.

In the following, the convergence order of numerical solutions computed with the presented adaptive platform is established for all the different strategies described above. First, the case of smooth flows is given, then the case of singular flows is exemplified by the case of a traveling discontinuity.

5.1. Smooth flow fields

For some Hessian-based methods, i.e., the $\mathbf{L}^\infty - \mathbf{L}^p$ approach of [22], an analysis is proposed for predicting the order of convergence to the continuous solution. In the goal-oriented method discussed in this paper, the adaptive state solution W_h generally does *not* converge to the continuous one W. However, in the case of regular solutions, the expression of the optimal error model indicates that the functional output does converges, and with a predictable order.

Smooth context with specified time step. Here, we are adapting the mesh with the purpose of reducing the spatial error. The space dimension is 3. In this case, we have established the following estimate:

$$\mathbf{E}(\mathbf{M}_{go}) = O\left(N_{st}^{-\frac{2}{3}}\right) \quad \text{as } N_{st} \to \infty,$$

for the case of an adaptation at each time step (34) and also for the case of an adaptation with a fixed mesh at each sub-interval (38), therefore:

Lemma 5.1. *The modeled spatial error on functional for a specified time-step converges at second-order rate.*

Smooth context with Courant-based time step. According to the argument recalled above, we are adapting the mesh $\mathbf{M}(\mathbf{t})$ in order to, by the magic of Courant condition, reduce both space and time error. The space–time dimension is 4. Now, in this case, we have established the following estimate:

$$\mathbf{E}(\mathbf{M}_{go}) = O(N_{st}^{-\frac{1}{2}}) \text{as} N_{st} \to \infty,$$

for the case of an adaptation at each time step, (36), and also for the case of an adaptation with a fixed mesh at each sub-interval, (40), therefore:

Lemma 5.2. *The space–time modeled error on functional for Courant-based time step converges at second-order rate.*

5.2. Singular flow fields

An important advantage of mesh adaptation is its potential ability for addressing the case of solution fields involving (isolated) singularities such as discontinuities, etc. with a better mesh convergence. In this case, it can happen that the specified mesh density become locally infinitely large and the mesh size can be zero. In the unsteady case, the time step also becomes zero or too small. Not only the time advancing is stopped or too slow, but also the evaluation of the global effort (defined by the mesh size divided by the time step) becomes difficult to use. This can be avoided by forcing the mesh size to be larger than a given length. We do not consider this issue in this section. We propose a short analysis dealing with the case of a *prescribed time step* and suggesting a few rules for attempting to have second-order *spatial* convergence.

According to remarks of [31], second-order space convergence can *also* be obtained for singular case with discontinuities, assuming that the error on singularity is concentrated on a subset of zero measure of the computational domain to be discretized by the mesh. For simplicity, we consider only the error committed when we simply *interpolate* a given function on a mesh.

Steady flow with a shock. The $\mathbf{L}^1$ convergence of a piecewise linear interpolation is of second order far from the shock. It is only first-order on the shock. To reduce the error by a factor of four, mesh-size normal to the shock needs to be reduced by a factor of four. In several papers, as in [31], we have demonstrated that a good anisotropic mesh adaptation algorithm allows us to still get second-order convergence. This is made possible by imposing that the region near singularity, where mesh size should be four times smaller, can itself have its measure reduced by a factor four. Therefore, it is possible to apply the following convergence strategy (3D case):

- mesh size outside shock is divided by 2 in the three space directions and the number of nodes is multiplied by 8. Taking into account the second-order accuracy of the considered interpolation, this produce a local error that is 4 times smaller,
- the thickness of the region around the shock is divided by 4. In the new region around the shock, mesh size is divided by 2 parallel to the shock and by 4 normal to the shock. Although the density of nodes is multiplied by 16, the number of nodes is multiplied by 4, due to the reduction of thickness. Consequently, the overall number of nodes is essentially multiplied by 8 for an error divided by 4.

Lemma 5.3. *For a steady flow field involving a shock, interpolation error can be divided by a factor of 4 with a total number of nodes multiplied by 8, which expresses the second-order spatial convergence. This property does not only hold for anisotropic mesh adaptive interpolation, but also for anisotropic mesh adaptive algorithms for PDEs, see* [31].

Unsteady flow with a moving shock. The space-wise second-order capture of the singularity is well addressed by the spatial mesh adaptation: for each time level, we apply a metric-based anisotropic spatial mesh adaptation. Mesh size normal to the discontinuity will be reduced by a factor of 4 only in the vicinity of the discontinuity. In other directions near discontinuity and in any direction in other regions of the computational domain, the mesh size is divided by 2. In 3D, with this strategy, passing from N spatial degrees of freedom to $8N$ leads to a four times reduction of the spatial error. Unfortunately, the time step also needs to be reduced by a factor 4, leading to a total number of degrees of freedom multiplied by 32. Thus:

$$||u - u_{32N}||_{L^1(\Omega\times[0,T])} \leqslant \frac{1}{4}||u - u_N||_{L^1(\Omega\times[0,T])}. \tag{41}$$

By comparing $\frac{1}{4}$ with $32^{-\alpha/n}$, this time with $n = 4$ (space–time dimension), we get the following barrier convergence lemma:

Lemma 5.4. Barrier convergence order for time advancing discretizations. *For a time-advancing second-order flow solver coupled with an unsteady goal-oriented mesh adaptation at each time level applied to a traveling discontinuity, the space–time convergence rate is at most* $\alpha = 8/5$.

Remark 3. The limitation at rate $8/5$ applies to the cases where both smooth regions and discontinuities are present. In the case of a flow involving only discontinuities and no smooth region, the spatial mesh size can be bounded at $4N_{tot}$ and the second order can be recovered. In the case without discontinuity, the second order is automatically recovered because the time step is only reduced by a factor 2.

In fact, we are not only using a time advancing space–time mesh, but we are also considering the fixed-point mesh adaptation algorithm. The situation is (slightly) worse since we are working with meshes that are prismatic in time, that is, made of the product of a single spatial mesh by several time levels. This single spatial mesh needs to account for the *motion* of any singularity. If the singularity, for example a shock wave, moves during the simulation, then the region of the spatial mesh where the singularity progresses is the *union* of all singularity neighboring regions for any time level of the time interval. We call this region the trajectory of the discontinuity. In contrast to the steady case, this region is a *thick* region. At convergence, the measure of this thick region swept by the moving discontinuity is essentially proportional to its normal velocity times the time sub-interval width ΔT. For a single time sub-interval of fixed width, this measure cannot tend to zero, hence a large number of nodes is wasted. When we apply a process of convergence to the continuous, we divide by 4 the mesh size normal to discontinuity and by 2 in the other directions. With an unchanged time sub-interval width, the number of nodes in the trajectory of the discontinuity is 16 times larger and 8 times larger for the rest of the mesh. It is thus mandatory to decrease time sub-interval width ΔT by a factor 2 (i.e., to increase n_{adap} accordingly in the case where $\Delta T = T/n_{adap}$) to recover higher order convergence. In a rather artificial manner, we can analyze the spatial convergence order by considering only the mean number of nodes, quotient of total space–time number of nodes by the number of time steps: $N_{mean} = N_{st}/n_\tau$.

Lemma 5.5. Spatial convergence for the transient fixed-point mesh adaptation. *For an unsteady flow field with a moving shock, a necessary condition for second order convergence in space is that the time sub-interval width is divided by 2 when smooth region mesh size is divided by 2 in each of the three space directions and when singular region mesh size is divided by 2 in directions parallel to shock and by 4 in direction normal to shock. To synthesize, in the case of a uniform division into time sub-intervals of size* $\Delta T = T/n_{adap}$, *we expect to get a spatial second-order spatial convergence in* $\mathbf{L}^1([0,T];\mathbf{L}^1(\Omega))$ *by passing from* (N_{mean}, n_{adap}) *to* $(8N_{mean}, 2n_{adap})$.

The more realistic evaluation of space–time convergence order needs to account for the number or equivalently the size of the time step. According to Lemma 5.4, we cannot hope better than a convergence at order $8/5$. We divide the time step by 4, since the traveling discontinuity is also a timewise discontinuity. This increases by a factor 4 the space–time number of nodes which needs also to be increased by a factor 8 at each time level. We also need to divide the time sub-interval length ΔT by 2 for limiting the number of nodes in the vicinity of the discontinuity trajectory.

Lemma 5.6. Space–time convergence for the transient fixed-point mesh adaptation. *We consider a second-order flow solver coupled with an unsteady goal-oriented mesh adaptation applied to a 3D traveling discontinuity. We assume a uniform division of the simulation time frame into time sub-intervals of size* $\Delta T = T/n_{adap}$. *Then, we expect a space–time convergence in* $\mathbf{L}^1([0,T];\mathbf{L}^1(\Omega))$ *at order* $8/5$ *by dividing the time step by a factor 4:* $\tau \to \tau/4$ *and passing from* (N_{st}, n_{adap}) *to* $(32N_{st}, 2n_{adap})$.

Remark 4. Adapting for a time sub-interval instead of adapting at each time steps does not degrade the asymptotic convergence order which is a very good news. Nevertheless, such a series of adapted meshes is only space–time sub-optimal as the constant in the error term is larger than the adaptation at each time step.

Remark 5. Between sub-intervals, transfers of the solution fields from the previous mesh to the new one are necessary. The choice of the transfer operator has certainly some impact on the global accuracy (see for example [6]) together with how frequently it is applied. Reference [6] suggests to consider conservative interpolation for compressible flows.

6. From theory to practice

The continuous mesh adaptation problem takes the form of the following continuous optimality conditions:

$$\begin{aligned} &W \in \mathcal{V}, \forall \varphi \in \mathcal{V}, (\Psi(\mathcal{M}, W), \varphi) = 0 \quad \text{“Euler system”} \\ &W^* \in \mathcal{V}, \forall \psi \in \mathcal{V}, \left(\frac{\partial \Psi}{\partial W}(\mathcal{M}, W)\psi, W^*\right) = (g, \psi) \quad \text{“adjoint system”} \\ &\mathcal{M}(\mathbf{x}, t) = \mathcal{M}_{go}(\mathbf{x}, t) \quad \text{“adapted mesh”}. \end{aligned} \tag{42}$$

In practice, it remains to approximate the above three-field coupled system by a discrete one and then solve it. For discretising the state and adjoint PDE's, we take the spatial schemes introduced previously which are explicit Runge–Kutta time advancing schemes. Such time discretization methods have non-linear stability properties like TVD which are particularly suitable for the integration of system of hyperbolic conservation laws where discontinuities appear. Discretising the last equation consists in specifying the mesh according to a discrete metric deduced from the discrete states.

In order to remedy all the problematics relative to mesh adaptation for time-dependent simulations stated in the introduction, an innovative strategy based on a fixed-point algorithm has been initiated in [3] and fully developed in [2]. The fixed-point algorithm aims at avoiding the generation of a new mesh at each solver iteration which would imply serious degradation of the CPU time and of the solution accuracy due to the large number of mesh modifications. It is also an answer to the lag problem occurring when, from the solution at t^n, a new adapted mesh is generated at level t^n to compute next solution at level t^{n+1}. In that latter case, since the mesh is not adapted to the solution evolution between time levels t^n and t^{n+1}, the mesh is always late with respect to the physics. The fixed point approach has been successfully applied to bi-fluids three-dimensional problems [22], to a blast simulation in a three-dimensional city [2] and to moving bodies simulations [7]. The basic idea consists in splitting the simulation time frame $[0, T]$ into n_{adap} adaptation sub-intervals:

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

and to keep the same adapted mesh for each sub-interval. Consequently, the time-dependent simulation is performed with only n_{adap} different adapted meshes. The mesh used on each sub-interval is adapted to control the solution accuracy from t_{i-1} to t_i. We examine now how to apply this program.

6.1. Choice of the goal-oriented metric

The optimal adapted meshes for each sub-interval are generated according to analysis of Section 4.6. In this work, the following particular choice has been made:

- the Hessian metric for sub-interval i is based on a control of the temporal error in $\mathbf{L}^\infty$ norm:

$$\mathbf{H}^i_{\mathbf{L}^\infty}(\mathbf{x}) = \Delta t_i \max_{t \in [t_i, t_{i+1}]} \mathbf{H}(\mathbf{x}, t) = \Delta t_i \mathbf{H}^i_{\max}(\mathbf{x}),$$

- function $\tau : t \to \tau(t)$ is constant and equal to 1,
- all sub-intervals have the same time length Δt.

The optimal goal-oriented metric $\mathbf{M}_{go} = \{\mathbf{M}^i_{go}\}_{i=1,\ldots,n_{adap}}$ then simplifies to:

$$\mathcal{M}^i_{go}(\mathbf{x}) = N_{st}^{\frac{2}{3}} \left(\sum_{i=1}^{n_{adap}} \left(\int_\Omega (\det \mathbf{H}^i_{\max}(\mathbf{x}))^{\frac{1}{5}} d\mathbf{x} \right) \right)^{-\frac{2}{3}} (\Delta t)^{\frac{1}{3}} \left(\det \mathbf{H}^i_{\max}(\mathbf{x}) \right)^{-\frac{1}{5}} \mathbf{H}^i_{\max}(\mathbf{x}).$$

Remark 6. We notice that we obtain a similar expression of the optimal metric to the one proposed in [7], but it is presently obtained in a goal-oriented context and by means of a space–time error minimization.

6.2. Global fixed-point mesh adaptation algorithm

To converge the non-linear mesh adaptation problem, i.e., converging the couple mesh-solution, we propose a fixed-point mesh adaptation algorithm. The parameter N_{st} representing the total computational effort is prescribed by the user and will influence the size of all the meshes defined during the time interval. That is, to compute any metric fields $\mathbf{M}^i_{go}$, we have to evaluate a global normalization factor which requires the knowledge of all $\mathbf{H}^i_{\max}$. Thus, the whole simulation from 0 to T must

be performed to be able to evaluate all metrics $\mathbf{M}^i_{go}$. Similarly to [7], a global fixed point strategy covering the whole time-frame $[0,T]$, called *Global adjoint fixed-point mesh adaptation algorithm*, is considered:

```
//- Fixed-point loop to converge global space–time mesh adaptation
For j = 1,n_nptfx
  //- Solve state once to get checkpoints
  For i = 1,n_adap
  • W^j_{0,i} = ConservativeSolutionTransfer(H^j_{i-1}, W^j_{i-1}, H^j_i)
  • W^j_i = SolveStateForward(W^j_{0,i}, H^j_i)
  End for
  //- Solve state and adjoint backward and store samples
  For i = n_adap,,1
  • (W*)^j_i = AdjointStateTransfer(H^j_{i+1}, (W*_0)^j_{i+1}, H^j_i)
  • {W^j_i(k), (W*)^j_i(k)}_{k=1,n_k} = SolveStateAndAdjointBackward(W^j_{0,i}, (W*)^j_i, H^j_i)
  • |H_max|^j_i = ComputeGoalOrientedHessianMetric(H^j_i, {W^j_i(k), (W*)^j_i(k)}_{k=1,n_k})
  End for
  • C^j = ComputeSpaceTimeComplexity({|H_max|^j_i}_{i=1,n_adap})
  • {M^j_i}_{i=1,n_adap} = ComputeUnsteadyGoalOrientedMetrics(C^j, {|H_max|^j_i}_{i=1,n_adap})
  • {H^{j+1}_i}_{i=1,n_adap} = GenerateAdaptedMeshes({H^j_i}_{i=1,n_adap}, {M^j_i}_{i=1,n_adap})

  End for
```

Let us describe this algorithm sketched in Fig. 4. It consists in splitting the time interval $[0,T]$ into the n_{adap} mesh-adaptation time sub-intervals: $\{[t_{i-1},t_i]\}_{i=1,\dots,n_{adap}}$ 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 a out-of-core storage of all checkpoints, which are taken to be $\{\mathcal{W}_h(t_i)\}_{i=1,\dots,n_{adap}}$. Between each sub-interval, the solution is interpolated on the new mesh using the conservative interpolation of [6]. 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 $\mathbf{H}^i_{\max}$ required to generate new adapted meshes for each sub-interval. To this end, n_k Hessian metric sample are computed on each sub-interval and intersected [2] to obtain $\mathbf{H}^i_{\max}$. 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 $j=1,\dots,n_{ptfx}$ global fixed-point loop. Convergence of the fixed-point is obtained in typically 5 global iterations.

This mesh adaptation loop has been fully parallelized. The solution transfer, the solver and the Hessians computation have been parallelized using a p-thread paradigm at the element loop level [4]. As regards the computation of the metrics and the generation of the adapted meshes, we observe that they can be achieved in a decoupled manner once the complete simulation has been performed, leading to an easy parallelization of these stages. Indeed, if n_{adap} processors are available, each metric and mesh can be done on one processor.

6.3. Computing the goal-oriented metric

The optimal goal-oriented metric is a function of the adjoint state, the adjoint state gradient, the state time derivative Hessian and the Euler fluxes Hessians. In practice, these continuous states are approximated by the discrete states and derivative recovery is applied to get gradients and Hessians. The discrete adjoint state W^*_h is taken to represent the adjoint state W^*. The gradient of the adjoint state ∇W^* is replaced by $\nabla_R W^*_h$ and the Hessian of each component of the flux vector $H(\mathcal{F}_i(W))$ is obtained from $H_R(\mathcal{F}_i(W_h))$. ∇_R (resp. H_R) stands for the operator that recovers numerically the first (resp. second) order derivatives of an initial piecewise linear solution field. In this paper, the recovery method is based on the $\mathbf{L}^2$-projection formula. Its formulation along with some comparisons to other methods is available in [5].

7. Numerical experiments

The adaptation algorithms described in this paper have been implemented the CFD code `Wolf`. As regards meshing, goal-oriented mesh adaptation requires to update the surface mesh of Γ on which the functional is observed. This standpoint is needed in order to ensure a valid coupling between the volume mesh and the surface mesh. This implies to consider a more complex re-meshing phase. In our case, a local remeshing strategy has been considered. We use `Yams` [17] for the adaptation in 2D and `Feflo.a` [32] in 3D.

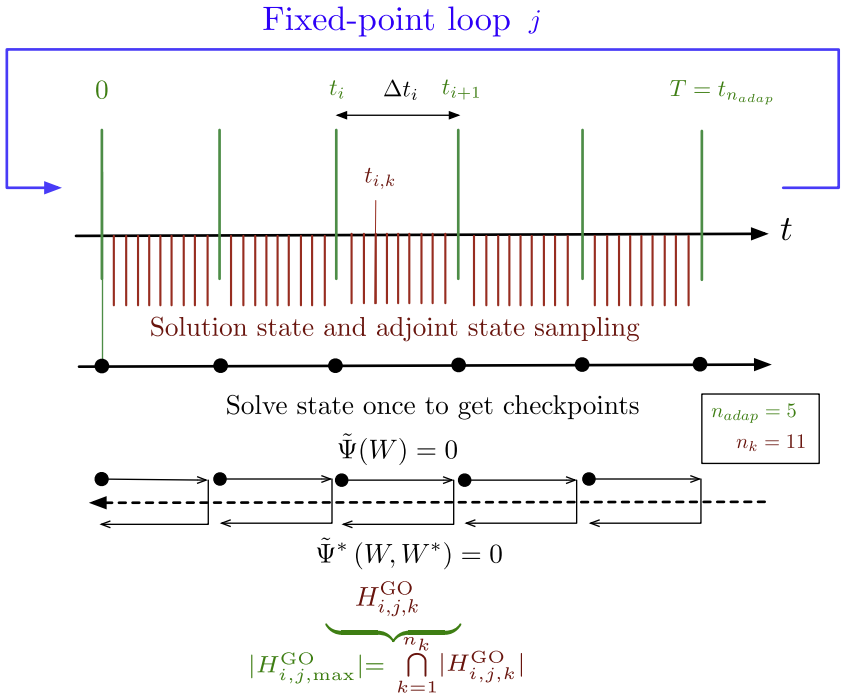

Fig. 4. Global Fixed-Point algorithm for unsteady goal-oriented anisotropic mesh adaptation.

7.1. 2D Blast wave propagation

We first apply the goal-oriented adaptive strategy to the example presented in Section 3.3. It consists in a 2D blast in a geometry representing a city. We recall that the cost function of interest j is the quadratic deviation of the ambient pressure on target surface S (see Fig. 1):

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

The simulation time frame is split into 30 time sub-intervals, i.e., 30 different adapted meshes are used to perform the simulation. Hence, 30 checkpoints are stored for the backward computation of the unsteady adjoint.

The resulting adjoint-based anisotropic adapted meshes are shown in Fig. 5. The corresponding density isolines are depicted in Fig. 6. These adapted meshes indubitably illustrate that, thanks to the unsteady adjoint, the mesh adaptation only focuses on shock waves that impact the observation region and ignores other area of the flow field. Therefore, waves traveling toward the observation are accurately captured whereas the rest of the flow is poorly computed. We also observe that once waves go beyond the target surface, the mesh is no more refined even if they continue traveling throughout the computational domain. Indeed, they do not impact anymore the functional.

Fig. 7 is a close up view of the adjoint-based adapted meshes corresponding to sub-intervals 8 and 15. The mesh anisotropy is clearly visible. The use of all variables in the metric definition does not affect the mesh anisotropy.

It is then quite interesting to compare the Hessian-based approach of [7] with our adjoint-based method. This comparison can be done by considering Figs. 5, 6 (top, right) and 8(top, right) and 8. It demonstrates how the adjoint defines an optimal distribution of the degrees of freedom for the specific functional, while it is clear that in this context the Hessian-based approach gives a non-optimal result for the evaluation of the functional but capture accurately the whole flow.

In conclusion, if an output functional of interest is provided then the reduction of the simulation number of degrees of freedom can be even more improved by considering a goal-oriented analysis instead of an Hessian-based methodology.

7.2. 2D acoustic wave propagation

Another typical example of pressure deviation propagating over long distances is linear acoustic. Linear acoustic waves usually refer either to a transient wave of bounded duration or to a periodic vibration. An important context in the study of these different kinds of waves is when we are interested only in the effect of the wave on a microphone occupying a very small part of the region affected by the pressure perturbation. Further simplifying, we can be interested in a single scalar measure of this effect, for instance the total energy E_{tot} received by the sensor during a given time interval. If the pressure

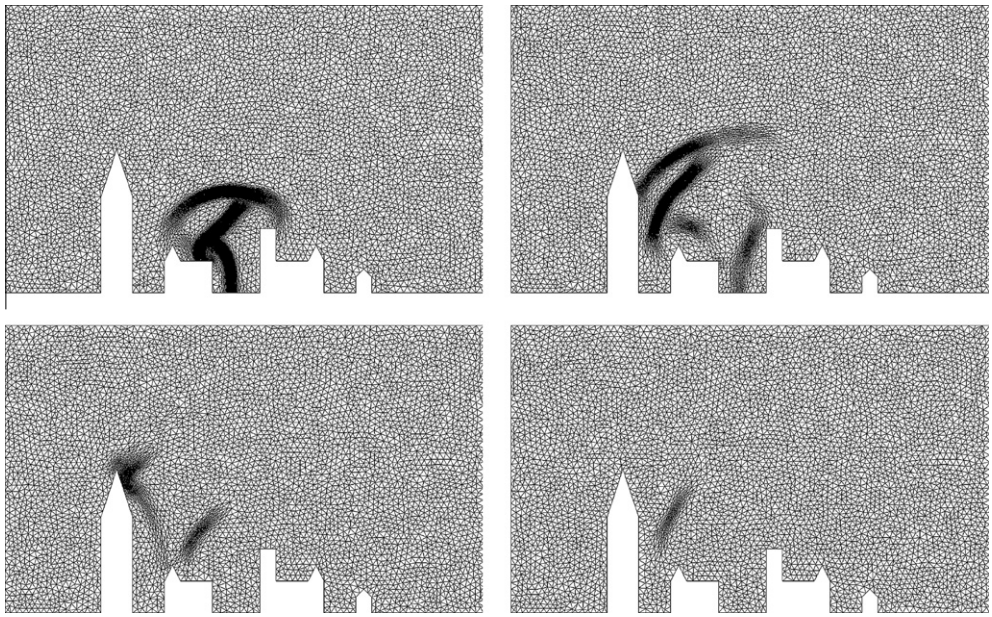

Fig. 5. 2D city blast adjoint-based adapted meshes evolution. From top to bottom and left to right, adapted meshes corresponding to sub-intervals 8, 15, 22 and 29 at non-dimensional time 1.2, 2.25, 3.3 and 4.35.

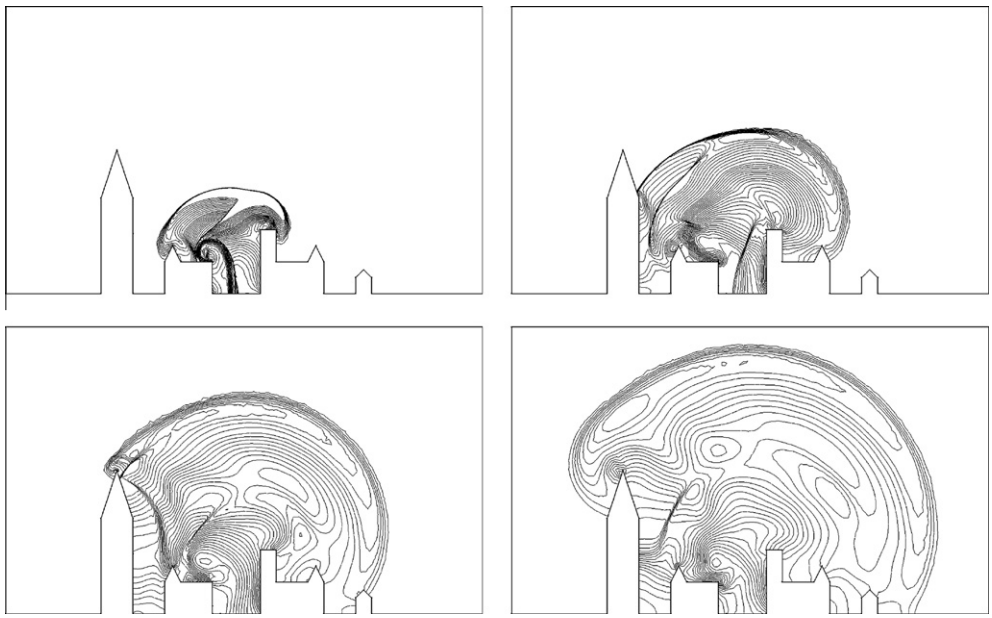

Fig. 6. 2D city blast adjoint-based adaptive solution state evolution. From top to bottom and left to right, density iso-lines corresponding to the end of sub-intervals 8, 15, 22 and 29 at non-dimensional time 1.2, 2.25, 3.3 and 4.35.

perturbation is emitted at a very long distance in an open and complex spatial domain, the numerical simulation of this phenomenon can be extremely computer intensive, if not impossible.

We consider a sound source located at the center-bottom of a rectangular domain. An acoustic source defined by $f = (0, 0, 0, r)$, where:

$$r = -Ae^{-Bln(2)\left[x^2+y^2\right]}Ampl \cdot cos(2\pi f)$$

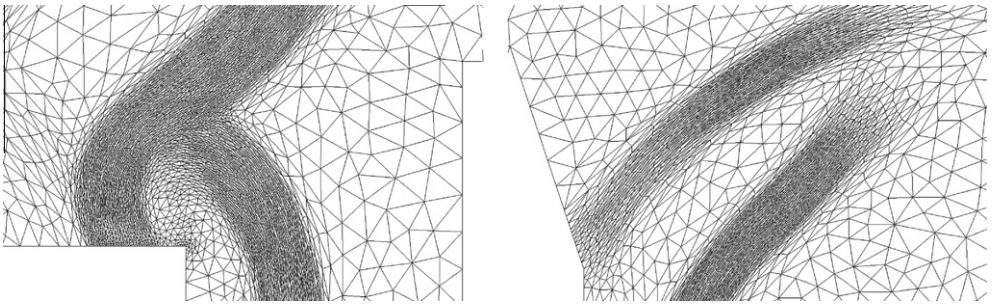

Fig. 7. Zoom in on adjoint-based adapted meshes corresponding to sub-intervals 8 (left) and 15 (right). Mesh anisotropy is clearly visible.

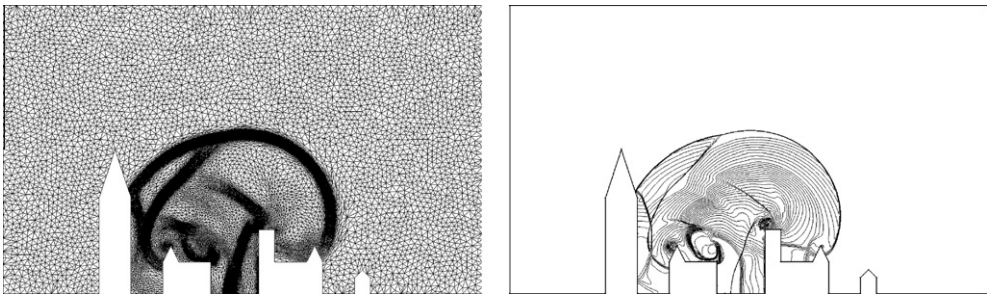

Fig. 8. Adapted mesh and corresponding density iso-lines for sub-interval 15 at non-dimensional time 2.25 obtained with the Hessian-based method of [7].

is added as a source term on the right-hand side of the unsteady compressible Euler Eqs. (9). Constants are $A = 0.01,\ B = 256,\ Ampl = 2.5$ and $f = 2$ is the waves frequency. We analyze the sound signal emitted by this source on a microphone M located at the center-top of the domain.

Goal-oriented mesh adaptation considers cost function:

$$j(W) = \int_0^T \int_M \frac{1}{2}(p(\mathbf{x},t) - p_0)^2 \mathrm{d}M\mathrm{d}t.$$

The simulation is split into 40 sub-intervals (thus 40 checkpoints are used) and 5 fixed-point iterations to converge the mesh adaptation problem. As expected, mesh adaptation reduces as much as possible mesh fineness in parts of the computational domain where accuracy loss does not influence the quality of sound prediction on the micro. This is illustrated in Fig. 9. Moreover, small perturbations of the waves on the uniform mesh are no more visible on the adapted anisotropic mesh, as shown in Fig. 10. Therefore, the solution computed on the adapted meshes is perfectly smooth. In Fig. 11 a zoom on the density field with associated uniform and anisotropic adapted mesh can be seen.

Accuracy study. An accuracy analysis of the proposed method are carried out on integrand $k(t)$ on the micro M:

$$k(t) = \int_M (p - p_0)\mathrm{d}M,$$

for different sizes of uniform and adapted meshes. $k(t)$ is plotted for three uniform meshes in Fig. 12. It illustrates that, for a rather coarse mesh of about 60 000 nodes, a small perturbation appears at the entrance of the micro. This perturbation diminishes with finer meshes. This spurious behavior is completely avoided by adaptive computations. Fig. 13 supports our affirmation and, as expected, our functional has a better prediction with adapted meshes.

In the case of higher frequencies, the choice of the mesh is even more crucial for a good prediction of the waves. Indeed, an increase of the frequency induces a drastic diminution of the spatial mesh size to preserve the computation accuracy:

$$\Delta x = \frac{\lambda}{n} = \frac{c}{nf},$$

where λ is the wave length, c the sound of speed and n the number of elements per wave length (usually $n \geqslant 5$). This leads to a huge increase in CPU's, therefore acoustic wave propagation in real-world applications remains a great numerical challenge. We simulate a higher frequency wave propagation on a uniform mesh composed of 11 7000 vertices and we compare it to a goal-oriented adaptive simulation. The obtained integrand $k(t)$ for both simulations is depicted in Fig. 14 which points out that in that case the choice of the mesh is even more crucial. Indeed, the proposed adaptive method is able to capture on the micro all the frequencies of the generated waves whereas a poor behavior is obtained with the uniform mesh.

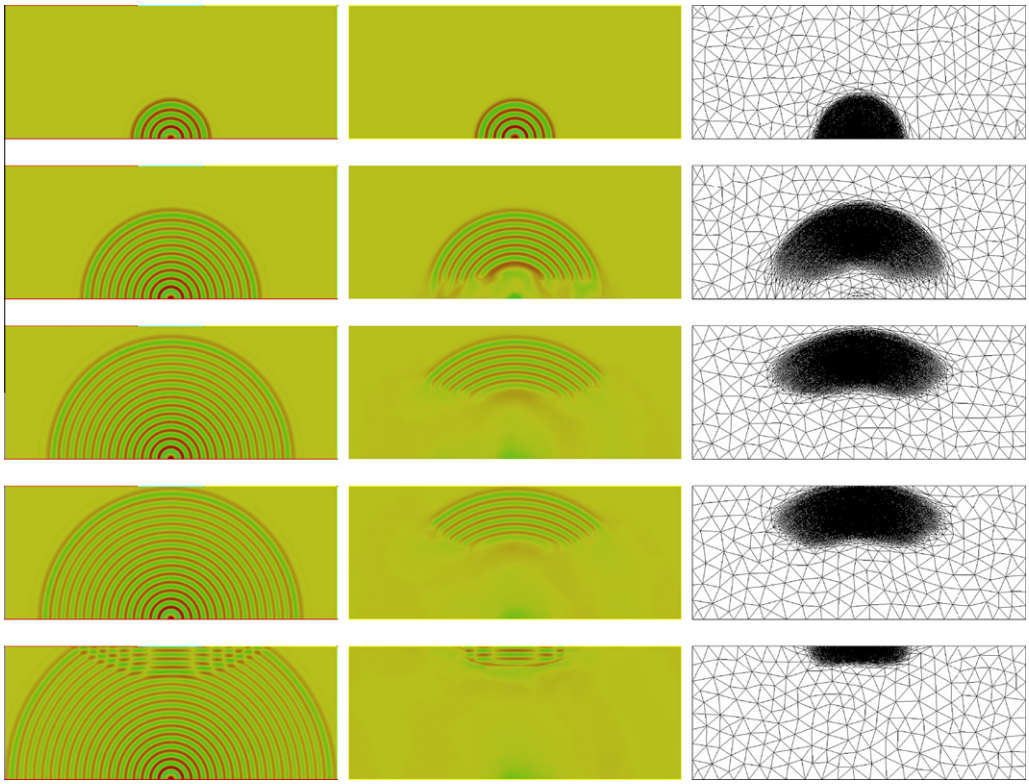

Fig. 9. Propagation of acoustic waves: density field evolving in time on a uniform mesh (left) and on adapted meshes (middle and right).

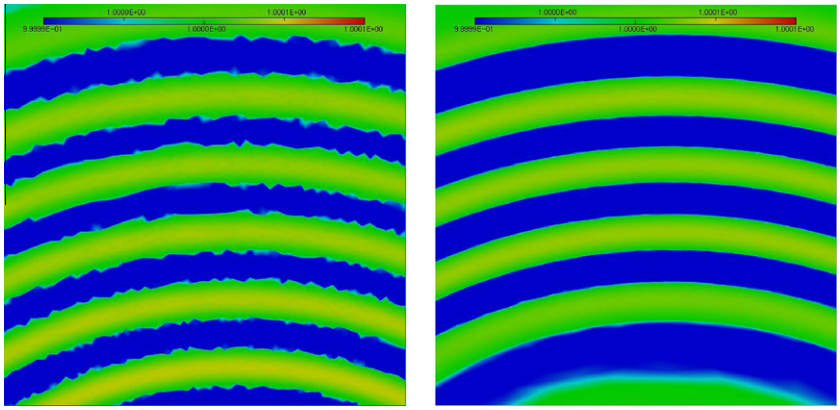

Fig. 10. Acoustic wave represented on uniform mesh (left) and on adapted mesh (right).

Convergence analysis. Now, a numerical convergence order analysis is done on the previous simulation with a wave frequency $f = 2$ in order to verify mathematical results of Section 5. For a mesh of N vertices, we denote by u_N and u_{exact} the approximate and exact solutions, respectively. The following relation holds:

$$u_N(\mathbf{x}, t) = u_{exact}(\mathbf{x}, t) + N^{-\frac{\alpha}{d}} u_1(\mathbf{x}, t) + o(N^{-\frac{\alpha}{d}}),$$

for d spatial dimension, α the convergence parameter to be found and u_1 the first normalized error term. Since α cannot be directly determined, an estimation is done on three meshes of different sizes: N_1, N_2 and N_3. Suppose u_{N_1}, u_{N_2} and u_{N_3} the corresponding numerical solutions, then we seek for α such that:

$$\frac{1 - \frac{N_2}{N_1}^{-\frac{\alpha}{d}}}{1 - \frac{N_3}{N_1}^{-\frac{\alpha}{n}}}, \approx \frac{u_{N_1} - u_{N_2}}{u_{N_1} - u_{N_3}} \tag{43}$$

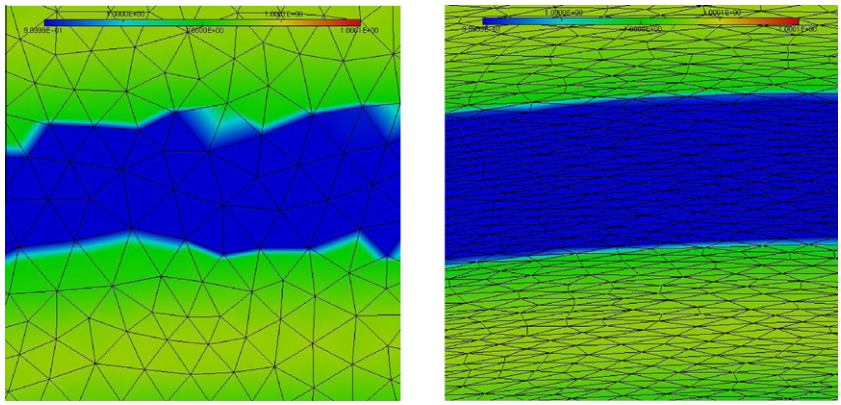

Fig. 11. Mesh visualization of uniform mesh (left) and adapted mesh (right).

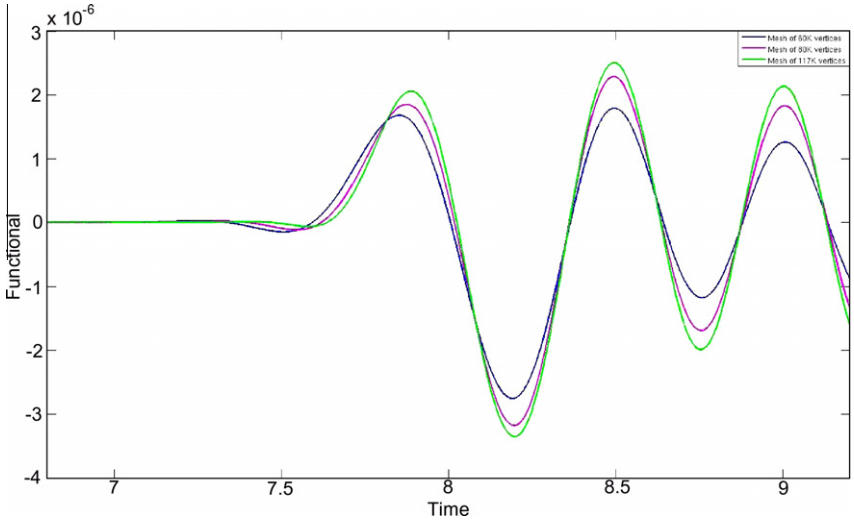

Fig. 12. Functional time integrand $k(t)$ on uniform meshes of size 60 K, 80 K and 117 K vertices for wave propagation with frequency $f = 2$.

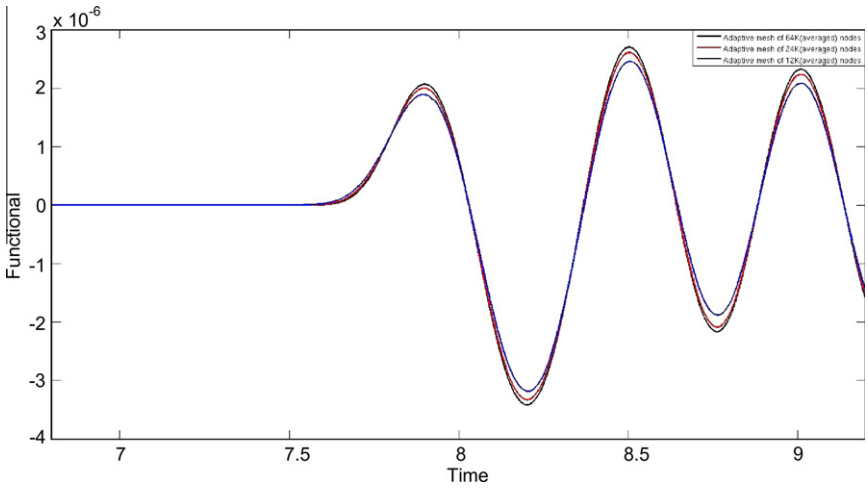

Fig. 13. Functional time integrand $k(t)$ on series of adaptive simulations with average meshes size of 12 K, 24 K and 64 K vertices for wave propagation with frequency $f = 2$.

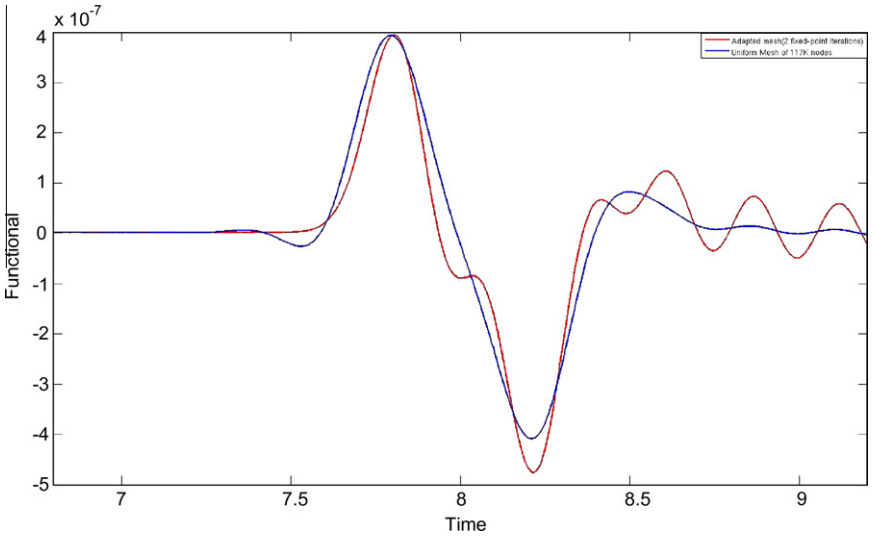

Fig. 14. Functional time integrand $k(t)$ for high frequency wave propagation with $f = 4$ for adaptive (red curve) and uniform mesh (blue curve) simulations. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

with dimension $d = 2$ in our case. Furthermore, we make the assumption that N_1 represent the higher number of vertices and N_3 the lowest one.

We solve Eq. (43) for both uniform and adapted meshes simulations presented above. The considered values for u_{N_i}, with $i = 1,\ 2,\ 3$, are the first maximal value of integrand $k(t)$ which are plotted in Figs. 12 and 13 for uniform meshes and for adaptive simulations, respectively. Table 1 summarizes the data collection: the convergence order is found to be 0.6 for uniform meshes and 1.98 for the adaptive simulations.

7.3. 3D blast wave propagation

Finally, the last example consider a purely three-dimensional blast problem in a complex geometry representing a city. The city size is 85 $m \times 70\ m \times 70\ m$. In this simulation, shock waves interact with each other and are reflected by buildings. The city geometry is the same as in [2]. Initially, the ambient air is at rest: $W_{air} = (1, 0, 0, 2.5)$. A "blast-like" initialization is considered inside a half-sphere of radius 2.5 m around position $(42, 53, 0)$: $W_{blast} = (10, 0, 0, 250)$. Cost function j is again the quadratic deviation from ambient pressure on target surface Γ which is composed of one building for simulation 1 or two buildings for simulation 2, see Fig. 15

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

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. Both simulations consider the same space–time complexity equal to 1.2 million.

The resulting adjoint-based anisotropic adapted meshes (surface and volume) for both simulations at sub-interval 10, 15 and 20 are shown in Figs. 16 and 17. 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

Table 1
Mesh convergence for the time-dependent pressure deviation on observation area.

Mesh	First observed maximum	Convergence order
Uniform mesh 60 K nodes	$1.67578e-06$	
Uniform mesh 80 K nodes	$1.84838e-06$	**0.6**
Uniform mesh 117 K nodes	$2.05461e-06$	
Adapted meshes 12 K (average) nodes	$1.89061e-06$	
Adapted meshes 24 K (average) nodes	$1.99895e-06$	**1.98**
Adapted meshes 64 K (average) nodes	$2.06722e-06$	

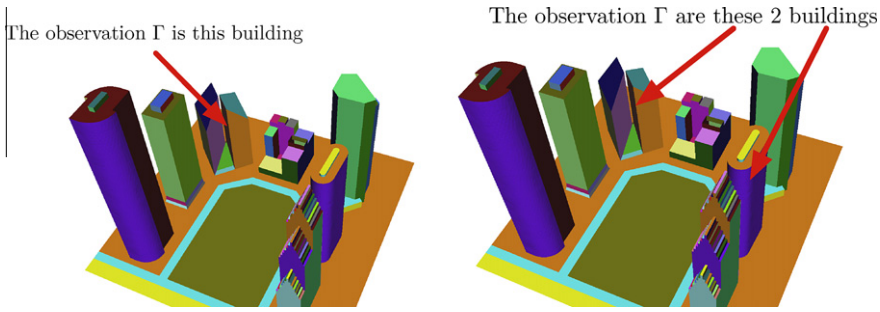

Fig. 15. 3D city test case geometry and location of target surface Γ composed of one building for simulation 1 (left) or two buildings for simulation 2 (right).

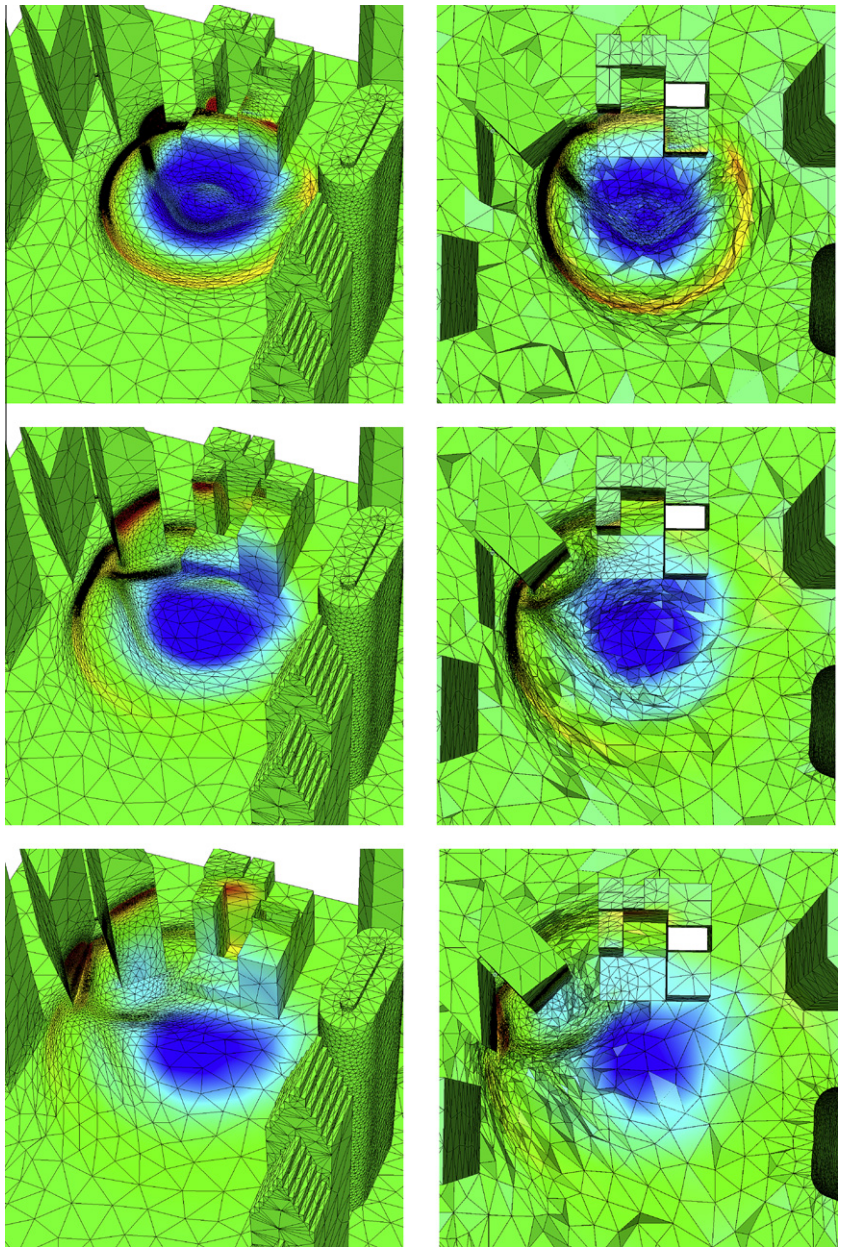

Fig. 16. 3D blast wave propagation: simulation 1. Adjoint-based anisotropic adapted surface (left) and volume (right) meshes at sub-interval 10, 15 and 20 and corresponding solution density at non-dimensioned time 5, 7.5 and 10.

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 of simulation 1 and 2 for sub-intervals 1, 5, 10 and 20 in Tables 2 and 3, respectively. Where nv, nt, nf are the number of vertices, tetrahedra and triangles and h the mesh size. If for sub-interval 1 one million vertices is needed, only forty thousands vertices are used for sub-interval 20. The required mesh size has been reduced by a factor 25 between the first and the twentieth sub-interval. In 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 4 where dozens of millions vertices are required to reach an accuracy of 30 cm.

As regards the amount of anisotropy achieved for these simulations, mesh anisotropy can be quantified by two different indicators: the anisotropic ratios and quotients. Deriving these quantities for an element relies on the fact that there always exists a unique metric tensor $\mathcal{M}_K$ for which this element is unit. Once $\mathcal{M}_K$ is computed, the anisotropic ratios and quotients associated with element K are simply given by

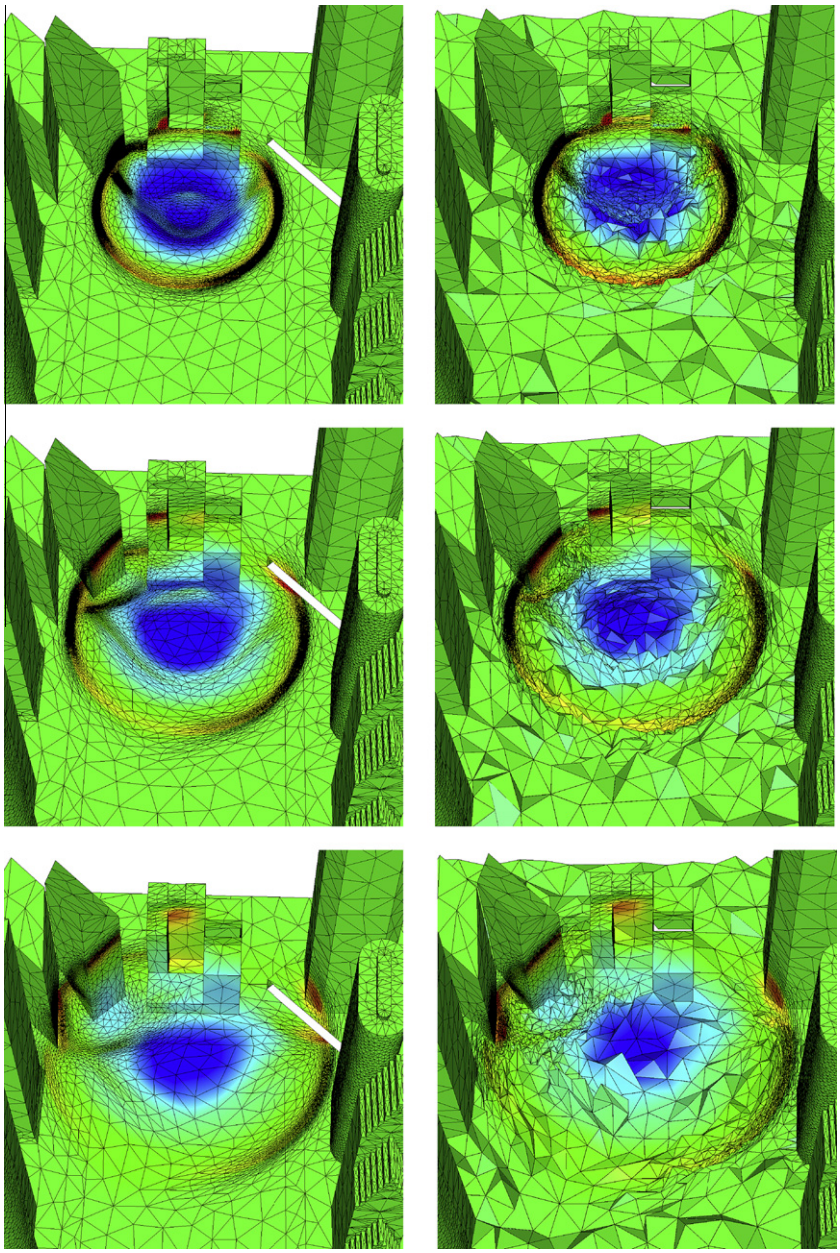

Fig. 17. 3D blast wave propagation: simulation 2. Adjoint-based anisotropic adapted surface (left) and volume (right) meshes at sub-interval 10, 15 and 20 and corresponding solution density at non-dimensioned time 5, 7.5 and 10.

Table 2
Mesh characteristics for simulation 1 of 3D blast wave calculation.

Iteration	nv	nt	nf	$\min h$	Ratio avg. (max)	Quotient avg. (max)
1	1058084	6177061	79862	1 cm	9(62)	63(2242)
10	249620	1427956	34318	1.2 cm	16(94)	197(5136)
15	72432	392970	24678	2.1 cm	12(76)	119(3789)
20	40297	205855	21980	4.8 cm	7(64)	39(3547)
Avg.	188357	1074777	28811			

Table 3
Mesh characteristics for simulation 2 of 3D blast wave calculation.

Iteration	nv	nt	nf	$\min h$	Ratio avg. (max)	Quotient avg. (max)
1	1051805	6139926	82858	1 cm	9(53)	67(1829)
10	233116	1327503	36376	1.2 cm	16(84)	203(4083)
15	77446	421024	26008	2.7 cm	12(85)	124(3654)
20	45500	235383	23072	6 cm	7(48)	51(1673)
Avg.	195355	1113492	31110			

Table 4
Uniform mesh characteristics for the 3D city geometry.

Uniform mesh	nv	nt	nf	Average h	Ratio	Quotient
1	10294136	60826207	862264	43 cm	2(12)	3(116)
2	33352859	198163572	2135032	29 cm	2(14)	3(165)

$$\text{ratio} = \sqrt{\frac{\min_i \lambda_i}{\max_i \lambda_i}} = \frac{\max_i h_i}{\min_i h_i} \quad \text{and} \quad \text{quo} = \frac{\max_i h_i^3}{h_1 h_2 h_3},$$

where $(\lambda_i)_{i=1,2,3}$ are the eigenvalues of $\mathcal{M}_K$ and $(h_i)_{i=1,2,3}$ are the corresponding sizes. The anisotropic ratio stands for the maximum elongation of a tetrahedron by comparing two principal directions. For both simulations an average anisotropic ratio between 7 and 16 is achieved. The anisotropic quotient represents the overall anisotropic ratio of a tetrahedron taking into account all the possible directions, we get a mean anisotropic quotient between 40 and 200. This quotient can be considered as a measure of the overall gain in three dimensions of *an anisotropic adapted mesh* as compared to *an isotropic adapted one*, here almost 100. The gain is of course even greater when compared to a uniform mesh.

8. Conclusion

We have designed a new mesh adaptation algorithm which prescribes the spatial mesh of an unsteady simulation as the optimum of a goal-oriented error analysis. This method specifies both mesh density and mesh anisotropy by variational calculus. Accounting for unsteadiness is applied in a time-implicit mesh-solution coupling which needs a non-linear iteration, the fixed point. In contrast to the Hessian-based fixed-point of [2,22] which iterates on each sub-interval, the new iteration covers the whole time interval, including forward steps for evaluating the state and backward ones for the adjoint. This algorithm has been successful applied to 2D and 3D blast wave Euler test cases and to the calculation of a 2D acoustic wave. Results demonstrate the favorable behavior expected from an adjoint-based method, which gives an automatic selection of the mesh regions necessary for the target output.

Several important issues for fully space–time computation have been addressed in this paper. 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.

Time discretization error is not considered in this study. Solving this question is not so important for the type of calculation that are shown in this paper, but can be of paramount impact in many other cases, in particular when implicit time advancing is considered. In a future work, the authors plan to consider a space–time error analysis in the context of the proposed method.

References

[1] P.-A. Absil, R. Mahony, R. Sepulchre, Optimization Algorithms on Matrix Manifolds, Princeton University Press, Princeton, NJ, 2008.
[2] F. Alauzet, P.J. Frey, P.L. George, B. Mohammadi, 3D transient fixed point mesh adaptation for time-dependent problems: application to CFD simulations, J. Comput. Phys. 222 (2007) 592–623.

[3] F. Alauzet, P.L. George, B. Mohammadi, P.J. Frey, H. Borouchaki, Transient fixed point based unstructured mesh adaptation, Int. J. Numer. Methods Fluids 43 (6-7) (2003) 729–745.
[4] F. Alauzet, A. Loseille, On the use of space filling curves for parallel anisotropic mesh adaptation, Proceedings of the 18th International Meshing Roundtable, 18, Springer, 2009, pp. 337–357.
[5] F. Alauzet, A. Loseille, High order sonic boom modeling by adaptive methods, J. Comput. Phys. 229: (2010) 561–593.
[6] F. Alauzet, M. Mehrenberger, P1-conservative solution interpolation on unstructured triangular meshes, Int. J. Numer. Methods Eng. 84 (13) (2010) 1552–1588.
[7] F. Alauzet, G. Olivier, Extension of metric-based anisotropic mesh adaptation to time-dependent problems involving moving geometries, in: 49th AIAA Aerospace Sciences Meeting and Exhibit, AIAA-2011-0896, Orlando, FL, USA, January 2011.
[8] V. Arsigny, P. Fillard, X. Pennec, N. Ayache, Log-Euclidean metrics for fast and simple calculus on diffusion tensors, Magn. Reson. Med. 56 (2) (2006) 411–421.
[9] M.J. Baines, Moving Finite Elements, Oxford University Press Inc., New York, NY, 1994.
[10] 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.
[11] Y. Belhamadia, A. Fortin, E. Chamberland, Three-dimensional anisotropic mesh adaptation for phase change problems, J. Comput. Phys. 201 (2004) 753–770.
[12] M. Berger, A Panoramic View of Riemannian Geometry, Springer Verlag, Berlin, 2003.
[13] M. Berger, P. Colella P, Local adaptive mesh refinement for shock hydrodynamics, J. Comput. Phys. 82 (1) (1989) 67–84.
[14] C.L. Bottasso, Anisotropic mesh adaption by metric-driven optimization, Int. J. Numer. Methods Eng. 60: (2004) 597–639.
[15] P.-H. Cournède, B. Koobus, A. Dervieux, Positivity statements for a Mixed-Element-Volume scheme on fixed and moving grids, Eur. J. Comput. Mech. 15 (7–8) (2006) 767–798.
[16] J. Dompierre, M.G. Vallet, M. Fortin, Y. Bourgault, W.G. 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, January 1997.
[17] P.J. Frey, `Yams`, a fully automatic adaptive isotropic surface remeshing procedure. RT-0252, INRIA, November 2001.
[18] P.J. Frey, F. Alauzet, Anisotropic mesh adaptation for CFD computations, Comput. Methods Appl. Mech. Eng. 194 (48–49) (2005) 5068–5082.
[19] M.B. Giles, N.A. Pierce, An introduction to the adjoint approach to design, Flow Turbul. Combust. 65 (2000) 393–415.
[20] M.B. Giles, E. Suli, Adjoint methods for PDEs: a posteriori error analysis and postprocessing by duality, Acta Numer. 11 (2002) 145–236.
[21] C. Gruau, T. Coupez, 3D tetrahedral unstructured and anisotropic mesh generation with adaptation to natural and multidomain metric, Comput. Methods Appl. Mech. Eng. 194 (48-49) (2005) 4951–4976.
[22] D. Guégan, O. Allain, A. Dervieux, F. Alauzet, An $l^{\infty} - l^{p}$ mesh adaptive method for computing unsteady bi-fluid flows, Int. J. Numer. Methods Eng. 84 (11) (2010) 1376–1406.
[23] F. Hecht, B. Mohammadi, Mesh adaptation by metric control for multi-scale phenomena and turbulence, AIAA Paper, 97-0859, 1997.
[24] T. Leicht, R. Hartmann, Error estimation and anisotropic mesh refinement for 3D laminar aerodynamic flow simulations, J. Comput. Phys. 229 (19) (2010) 7344–7360.
[25] X. Li, M.S. Shephard, M.W. Beal, 3D anisotropic mesh adaptation by mesh modification, Comput. Methods Appl. Mech. Eng. 194 (48–49) (2005) 4915–4950.
[26] R. Löhner, Adaptive remeshing for transient problems, Comput. Methods Appl. Mech. Eng. 75: (1989) 195–214.
[27] A. Loseille, F. Alauzet, Optimal 3D highly anisotropic mesh adaptation based on the continuous mesh framework, International Meshing Roundtable, 18, Springer, 2009, pp. 575–594.
[28] A. Loseille, F. Alauzet, Continuous mesh framework. Part I: Well-posed continuous interpolation error, SIAM J. Numer. Anal. 49 (1) (2011) 38–60.
[29] A. Loseille, F. Alauzet, Continuous mesh framework. Part II: Validations and applications, SIAM J. Numer. Anal. 49 (1) (2011) 61–86.
[30] A. Loseille, A. Dervieux, F. Alauzet, Fully anisotropic goal-oriented mesh adaptation for 3D steady Euler equations, J. Comput. Phys. 229: (2010) 2866–2897.
[31] A. Loseille, A. Dervieux, P.J. Frey, F. Alauzet, Achievement of global second-order mesh convergence for discontinuous flows with adapted unstructured meshes, in: 37th AIAA Fluid Dynamics Conference and Exhibit, AIAA-2007-4186, Miami, FL, USA, June 2007.
[32] A. Loseille, R. Löhner, Adaptive anisotropic simulations in aerodynamics, in: 48th AIAA Aerospace Sciences Meeting and Exhibit, AIAA-2010-169, Orlando, FL, USA, January 2010.
[33] C.C Pain, A.P. Humpleby, C.R.E. de Oliveira, A.J.H. Goddard, Tetrahedral mesh optimisation and adaptivity for steady-state and transient finite element calculations, Comput. Methods Appl. Mech. Eng. 190: (2001) 3771–3796.
[34] P.W. Power, C.C. Pain, M.D. Piggott, G.C. Norman, F. Fang, D.P. Marshall, A.J.H. Goddard, I.M. Navon, Adjoint goal-based error norms for adaptive mesh ocean modelling, Ocean Modell. 15 (2006) 3–38.
[35] A. Tam, D. Ait-Ali-Yahia, M.P. Robichaud, M. Moore, V. Kozel, W.G. Habashi, Anisotropic mesh adaptation for 3D flows on structured and unstructured grids, Comput. Methods Appl. Mech. Eng. 189 (2000) 1205–1230.
[36] R. Verfürth, A Review of A Posteriori Error Estimation and Adaptative Mesh-Refinement Techniques, Wiley Teubner Mathematics, New York, 1996.
[37] M. Wintzer, M. Nemec, M.J. Aftosmis, Adjoint-based adaptive mesh refinement for sonic boom prediction, in: AIAA 26th Applied Aerodynamics Conference, AIAA-2008-6593, Honolulu, HI, USA, August 2008.