INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2000; **32**: 725–744

Anisotropic mesh adaptation: towards user-independent, mesh-independent and solver-independent CFD. Part I: general principles

Wagdi G. Habashi[a,*], Julien Dompierre[a,1], Yves Bourgault[a,2], Djaffar Ait-Ali-Yahia[a,3], Michel Fortin[b] and Marie-Gabrielle Vallet[c,1]

[a] *CFD Laboratory, Department of Mechanical Engineering, Concordia University, 1455 de Maisonneuve Blvd. W., Montreal, Quebec, Canada, H3G 1M8*
[b] *Département de Mathématiques et Statistique, Université Laval, Ste-Foy, Quebec, Canada, C1K 7P4*
[c] *Industrial Materials Institute, NRC, 75 Boul. de Mortagne, Boucherville, Quebec, Canada, J4B 6Y4*

SUMMARY

The present paper is the lead article in a three-part series on anisotropic mesh adaptation and its applications to structured and unstructured meshes. A flexible approach is proposed and tested on two-dimensional, inviscid and viscous, finite volume and finite element flow solvers, over a wide range of speeds. The directional properties of an interpolation-based error estimate, extracted from the Hessian of the solution, are used to control the size and orientation of mesh edges. The approach is encapsulated into an edge-based anisotropic mesh optimization methodology (MOM), which uses a judicious sequence of four local operations: refinement, coarsening, edge swapping and point movement, to equi-distribute the error estimate along all edges, *without any recourse to remeshing*. The mesh adaptation convergence of the MOM loop is carefully studied for a wide variety of test cases. The mesh optimization generic coupling of MOM with finite volume and finite element flow solvers is shown to yield the same final mesh no matter what the starting point is. It is also shown that on such optimized meshes, the need for computational fluid dynamics (CFD) stabilization artifices, such as upwinding or artificial viscosity, are drastically reduced, if not altogether eliminated, in most well-posed formulations. These two conclusions can be considered significant steps towards mesh-independent and solver-independent CFD. The structure of the three-part series is thus, 1, general principles; 2, methodology and applications to structured and unstructured grids; 3, applications to three-dimensional flows. Copyright © 2000 John Wiley & Sons, Ltd.

KEY WORDS: edge-based error estimator; *a posterori* error estimation; structured and unstructured grids; mesh adaptation; spring analogy; mesh–solver coupling

---

* Correspondence to: CFD Laboratory, Department of Mechanical Engineering, Concordia University, 1455 de Maisonneuve Blvd. W., Montreal, Quebec, Canada, H3G 1M8.
[1] Current address: CERCA, Montreal, Canada.
[2] Current address: University of Ottawa, Ottawa, Canada.
[3] Current address: Pratt & Whitney Canada, Montreal, Canada.

1. INTRODUCTION

Finite volume and finite element methods are the most commonly used computational fluid dynamics (CFD) discretization methods due to their flexibility in constructing structured and unstructured meshes over arbitrary domains. While the debate over their relative merits still rages, and will certainly continue for a while, the lines between the two approaches are constantly blurring, with a fashionable compromise being the use of finite element approximations for diffusion terms and finite volume approximations for convective terms, with a distant alternative being their combination into control volume finite element formulations. With these points not yet conclusively settled, both camps have spent feverish, and perhaps inordinate, effort in the last two decades on algorithmic refinements, giving fleeting thought to the suitability of the meshes on which these constantly perfected algorithms are being applied. It is not exaggerated to use the analogy of teams perfecting a Formula 1 racing tire to only be ultimately able to use it on a golf cart.

Simultaneously, progress in the automatic generation of meshes has been steady. Suitable, but not necessarily appropriate, meshes can be generated for almost any two- or three-dimensional domains. These meshes, no matter how much experience is embedded into their generation, remain intuitive and heuristically defined. It is not unexpected for the results of a numerical simulation to strongly depend on the mesh chosen and it is not always possible, nor feasible, to demonstrate mesh-independent results for a given problem strictly via mesh refinement. At the same time, it is not difficult to illustrate via mesh adaptation that the most appropriate mesh, for a fixed geometry, will violently change depending on flow conditions, such as Reynolds number, Mach number, steadiness, inlet conditions, etc. If only because of this observation, it should be evident that one should not dissociate the mesh generation effort from the flow solver. Thus, the future in mesh generation is bound to include a tighter coupling of the mesh with the solution, with the consequence that the part of the CFD cycle spent in mesh generation can be shortened and made easier to automate, as the initial mesh generation would only be considered an initial step in a more global cycle.

For both structured and unstructured meshes, current mesh adaptation strategies suffer from several problems, not least among them is the fact that they often lead to an uncontrolled increase in the number of mesh points, in their ultimate search for higher precision or in the search for a more uniform error distribution. These refinement/coarsening and redistribution methods produce nearly isotropic meshes since their aim is to make the length scales of each element essentially the same in all directions [1–3]. It must be observed, however, that CFD simulations are highly demanding because they are asked to resolve physical features such as recirculations, flow separation and disparate gradient phenomena, such as in boundary layers and shocks. Most of these phenomena, however, are characterized by regions of steep gradients of the flow variables, embedded in or adjacent to regions where these variables vary much more smoothly. Mesh adaptation procedures ought, therefore, to take advantage of these highly directional flow features in order to avoid the need for a fine meshing of the entire computational domain.

In terms of error estimation, recent years have seen a rapid development of adaptation methods based on *a posteriori* estimators: with a solution u_h computed on a given mesh, one

wants to measure its precision, namely estimate the error with respect to the exact, but unknown, solution. Such an error measure would then be used to alter either the mesh or the discretization methodology to move towards a target precision level. It is important here to stress two points. First, such a process is seldom carried out to the end, i.e. mesh adaptation is often used only to improve a solution and lower its error, but seldom is it used to guarantee the final uniformity of the error level. Second, it must be emphasized that in the broader context, adaptation does not necessarily mean refinement. In other words, a properly adapted mesh to a given level of accuracy may be reached for structured grids without changing the number of grid points from an original one (i.e. it is not necessary to introduce hanging nodes) and with even fewer points in the case of unstructured meshes, as will be demonstrated in this series.

Along these lines, a directional approach was suggested by Peraire *et al.* [4], which consists in constructing anisotropic meshes with biased resolution along rapidly changing error estimate directions. They used an adaptive remeshing procedure for two-dimensional inviscid flows that incorporated directional stretching of triangular meshes. Such anisotropic meshes can also be produced by coupling a mesh movement strategy with local isotropic refinement (r–h method) [5]. Kornhuber *et al.* [6] proposed an anisotropic strategy based on directed refinement of pairs of triangular elements to resolve boundary layers. More recently, in Vallet [7], Fortin *et al.* [8–10] and Castro-Díaz *et al.* [11–13], a metric was used as a measure of error, coupled to an r–h strategy, to achieve directionally adapted unstructured meshes having high aspect ratios.

The present series of papers will, in a logical and sequential way, demonstrate that a cost-effective approach is the construction of directionally adapted, or anisotropic, meshes. With a properly chosen *a posteriori* error estimator, having strong directional properties, the mesh adaptation procedure will be shown to yield more appropriate meshes. Thus, the procedure can either be used in a post-processing spirit to simply improve a solution or, alternately, if coupled to the CFD solver in a continuous solver–adaptation loop, can be shown to yield optimized meshes.

The label optimized meshes, rather than optimal meshes, will be used in order to avoid the false impression that these are the best meshes that can be used, since they only are the best meshes used according to the selected error estimator. It will be shown that widely different initial meshes (ranging from extra coarse to extra fine, including counter-intuitive meshes) will converge through the optimization procedure to a statistically identical final mesh. It will also be shown that on such an optimized mesh, algorithmic refinements embedded into the flow solver will be diminished in impact, even leaving a fighting chance to first-order CFD methods.

The aim here, therefore, will be to describe general tools for structured and unstructured meshes, the latter including, besides mesh movement, refinement, coarsening and reconnection. Numerical results will be given to illustrate the range of capabilities of the method, while a more extensive validation will be presented in the following parts of this three-paper series, the second part dealing mainly with structured and unstructured grid issues and the third with three-dimensionality and applications respecting CAD integrity.

2. INTERPOLATION-BASED ERROR ESTIMATORS

Sharp estimates of the error, with an effectivity index close to unity, are usually difficult to derive for highly complex problems and/or expensive to evaluate. Thus, one is led to accept less precise but easily computable estimates as the basis for adaptive improvement, since the error estimate serves only as an indication of relative error between successive meshes, or approximation orders, and its calculation should not be allowed to take more than a small percentage of the overall solution time.

In the present section, an efficient and simple error estimator is derived using finite element interpolation theory. For linear elements, it is well known from Lagrange's error formula that the error term is proportional to the second derivative. For the sake of simplicity, the derivation of the error estimate will be illustrated for a one-dimensional problem and then generalized to the two-dimensional case. If the solution $g(x)$ is approximated by $g^h(x)$ with piecewise linear interpolation, as shown in Figure 1, a local approximation error e_E can be defined over an element E to be

$$e_E(\bar{x}) = g(\bar{x}) - g_E^h(\bar{x}), \tag{1}$$

where $\bar{x}$ belongs to the interval $[0, h_E]$.

The approximate solution, g_E^h, may be expressed as a function of its nodal values in the form

$$g_E^h(\bar{x}) = \left(1 - \frac{\bar{x}}{h_E}\right) g_I + \frac{\bar{x}}{h_E} g_{I+1}, \tag{2}$$

with the origin of $\bar{x}$ placed at node I. The elemental error at any point $\bar{x}$ is classically

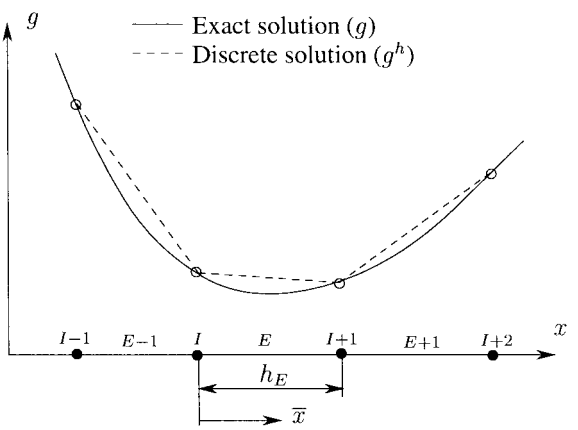

Figure 1. Approximate solution by a discrete method.

$$e_E(\bar{x}) = \left(\frac{\bar{x}^2}{2} - \frac{\bar{x}h_E}{2}\right) g''(\xi), \tag{3}$$

which is the departure of a quadratic interpolation from a linear one. Here, ξ is some point within the element E. It is straightforward to show that the maximum of the error on the element is given by

$$e_{\max} = \frac{h_E^2}{8} \left| \frac{\mathrm{d}^2 g^h}{\mathrm{d}x^2}(\xi) \right|_E. \tag{4}$$

Thus, the interpolation error for this one-dimensional problem is proportional to the product of the second derivative and the square of the characteristic length of the element, h_E.

An optimized mesh can thus be defined as a mesh for which the maximum error estimate is equi-distributed over all elements, i.e.

$$h_E^2 \left| \frac{\mathrm{d}^2 g^h}{\mathrm{d}x^2} \right|_E = C, \tag{5}$$

where C denotes a user-specified tolerance (positive constant).

To extend this adaptation criterion to the two-dimensional case, the second derivative of g^h in the direction of a given unit vector $\boldsymbol{V}$ is computed as follows:

$$\frac{\partial^2 g^h}{\partial \boldsymbol{V}^2} = \boldsymbol{V}^T \boldsymbol{H} \boldsymbol{V}, \tag{6}$$

where $\boldsymbol{H}$ represents the Hessian matrix of g^h and is expressed as

$$\boldsymbol{H} = \begin{pmatrix} \dfrac{\partial^2 g^h}{\partial x^2} & \dfrac{\partial^2 g^h}{\partial x\, \partial y} \\ \dfrac{\partial^2 g^h}{\partial y\, \partial x} & \dfrac{\partial^2 g^h}{\partial y^2} \end{pmatrix}. \tag{7}$$

Let us consider an edge E of the triangulation and let τ be the unit tangential vector and l be the length of that edge. The maximum interpolation error along the edge can be estimated again by

$$e_{\max} = \frac{l^2}{8} \left| \frac{\partial^2 g^h}{\partial \tau^2}(\xi) \right|_E. \tag{8}$$

and one may use the same equi-distribution principle as in the one-dimensional case, i.e. the error should be made the same on all edges.

We shall also develop this idea with a more geometric analogy. The error estimate on an edge can be thought of as the length of this edge in a Riemannian metric deduced from the Hessian matrix $\boldsymbol{H}(x)$. Indeed, the symmetric Hessian can be diagonalized as

$$\boldsymbol{H} = \boldsymbol{R}\boldsymbol{\Lambda}\boldsymbol{R}^T, \tag{9}$$

where $\boldsymbol{\Lambda}$ is the diagonal matrix of the eigenvalues of $\boldsymbol{H}$ and $\boldsymbol{R}$ is the matrix of the eigenvectors. In order to obtain a symmetric positive-definite matrix, the Hessian $\boldsymbol{H}$ is modified into the metric $\boldsymbol{M}$ by taking the absolute value of its eigenvalues [7]. This results in

$$\boldsymbol{M} = \boldsymbol{R}|\boldsymbol{\Lambda}|\boldsymbol{R}^T. \tag{10}$$

From here on, we shall use the $\boldsymbol{M}$ length of any vector $\boldsymbol{V}$ to be defined by

$$\|\boldsymbol{V}\|_{\boldsymbol{M}}^2 = \boldsymbol{V}^T\boldsymbol{M}\boldsymbol{V}. \tag{11}$$

The matrix $\boldsymbol{M}$ will be a function of the co-ordinates x, so that we shall have a Riemannian space. The goal is to rebuild the mesh so that all triangles are equilateral in this space. It has been shown in the work of d'Azevedo and Simpson [14,15] that this goal is not, in general, attainable while remaining in a planar domain: a general Riemannian space is curved. However, a reasonable compromise is arrived at through a simple heuristic procedure. In this new context, Equation (8) is slightly modified: let $[x_I, x_J]$ be an edge of the triangulation, we thus write

$$e = (x_I - x_J)^T\boldsymbol{M}(x_I - x_J) = C. \tag{12}$$

An optimal mesh is thus defined as the mesh in which the length of all edges, in the defined metric $\boldsymbol{M}$, is equal to $\sqrt{C}$; the equi-distribution principle is applied to e.

Since $\boldsymbol{M}$ is a function of the space co-ordinates, Equation (12) must be modified one step further. We must evaluate

$$e(x_I - x_J) = \int_0^1 \sqrt{(x_I - x_J)^T\boldsymbol{M}(l)(x_I - x_J)}\,\mathrm{d}l. \tag{13}$$

This integral will be approximated by some quadrature rule. In practice, $\boldsymbol{M}(x)$ will be computed using a procedure described below and stored on a (fixed) background mesh. The value of $\boldsymbol{M}(x)$ at any position of the domain will be interpolated on this background mesh whenever needed during an adaptive loop, defined as the interval between two solutions of the system of equations. With this, the edge-based error estimate can then be numerically evaluated from Equation (13) for each edge of an element.

Remark 1

In Peraire's work [4], a related procedure has been developed. The L^2 error is equi-distributed in the direction of the eigenvectors, where $h = h_k$, with $k = 1, 2$, are two local spacings and $\boldsymbol{V} = \boldsymbol{V}_k$ are the two unit eigenvectors of the $\boldsymbol{M}$ matrix. Accordingly, the optimal mesh criterion (12) simplifies to

$$h_k^2|\lambda_k| = C \quad \text{with } k = 1, 2. \tag{14}$$

This equation serves to compute two local spacings, $h_k = \sqrt{C/|\lambda_k|}$, in two orthogonal directions, at any point in the domain. Then, a new adapted mesh is regenerated based on these parameters and principal directions of $\boldsymbol{M}$.

2.1. Computing the Hessian

There remains the question of obtaining, from a piecewise-linear finite element approximation u_h, an estimate of the Hessian matrix. There exists many possibilities to do so, and the simplest and most precise one is to look for a piecewise-linear $\boldsymbol{H}(x) = (h_{ij}(x))$ defined through a weak formulation, i.e.

$$\int_\Omega h_{ij}\varphi_h \, \mathrm{d}x = -\int_\Omega \frac{\partial u_h}{\partial x_i}\frac{\partial \varphi_h}{\partial x_j} \, \mathrm{d}x, \tag{15}$$

where φ_h is a test function (for the moment, vanishing on the boundary). The left-hand-side of Equation (15) implies a mass matrix that can be lumped, yielding for a vertex I

$$h_{ij}(I) = -\left(\int_\Omega \frac{\partial u_h}{\partial x_i}\frac{\partial \varphi_I}{\partial x_j} \, \mathrm{d}x\right)\left(\int_\Omega \varphi_I \, \mathrm{d}x\right)^{-1}, \tag{16}$$

where Ω_I is the set of elements surrounding I and φ_I is the associated linear shape function. This enables the computation of h_{ij} (I) for all interior nodes. At boundary nodes, boundary integrals containing $\partial u_h/\partial n$ should appear in Equation (15), but ignoring them is equivalent to assuming $\partial^2 u_h/\partial n^2 = 0$ (as in a free-spline approximation in a one-dimensional problem). In practice, good results are obtained by imposing $\partial h_{ij}/\partial n = 0$ on the boundary; this condition being obtained at boundary nodes through a finite difference formula.

2.2. Equi-distributing the error

With the estimate (13) of the interpolation error, one can evaluate the distribution of the error over all edges. The remaining question is to determine the mesh that will minimize this error estimate. It is often logically argued that the error will be minimized for a given number of mesh points when it has been equi-distributed. It is also usual to define the error estimate over an element (triangle or rectangle) in the same way that it has been defined here over an edge and to adapt according to the fact that

> 'the error estimate is equi-distributed when the error estimate is the same for all the *elements* of the mesh.'

This assumption, however, usually leads to isotropic adaptation with, for example, a shock wave being detected and refined, but 'equally' in all directions (see Figure 2).

The present approach bases the adaptation process on the following definition of equi-distribution of the error estimate:

> 'the error estimate has been equi-distributed when the error estimate is the same for all *edges* of a mesh.'

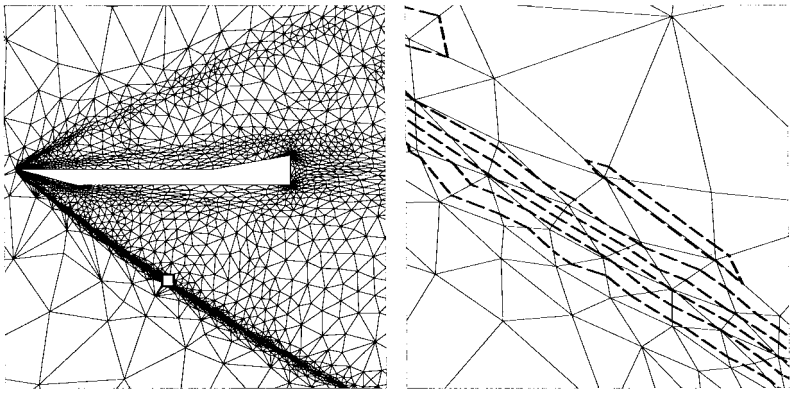

Figure 2. Adapted mesh with equi-distribution of the error estimate over the triangles and zoom on the shock with isovalues of the Mach number.

This approach yields highly stretched meshes (see Section 4) because information on each edge has been made *directional*.

Intuition and practice make one believe that a good, or at least a smooth, mesh should consist of equilateral triangles (or squares), this being the only recourse if nothing is known about the characteristics of the problem at hand. Making the error estimates equal on all edges makes the mesh consist of equilateral triangles or squares in the Riemannian metric defined by the Hessian of the solution.

3. ANISOTROPIC MESH OPTIMIZATION METHODOLOGY (MOM)

There are many commercial and free-ware mesh generators to build a mesh from some description of the domain boundaries (see http://www.tfd.chalmers.se/CFD_Online for an almost complete list of such applications), some of which even use the error criterion presented here. We were, however, to the best of our knowledge, the first to suggest another approach: from an existing mesh and from some description of the domain boundaries, to *modify the mesh instead of creating a new one* as was done in, for example, Vallet [7] and Catro-Díaz *et al*. [13], and to do so by combining four different local mesh modification strategies driven by the same directional error estimate.

There are two major advantages to proceed in such a manner. First, creating a new mesh is costly and the overhead remains constant, even as the coupled mesher–solver is approaching a converged solution on a nearly adapted mesh, while modifying an existing mesh may be inexpensive, especially when there are few modifications between the previous mesh and the adapted one. Second, our suggested approach is better suited for two of the main adaptation tasks. At the beginning of an adaptation loop, the working mesh is usually the background mesh, so location of nodes on the background is well initialized. Then, from loop to loop, the

last adapted mesh is employed as the background mesh so that it is less likely to miss an important feature of the metric tensor field when numerical integration is used to compute integral (13). For example, problems could arise if one wanted to compute this integral on a long edge crossing a thin shock, which could then be inadvertently skipped if the integration points do not happen to fall within the shock thickness or spread.

To build an optimized mesh with all edges having approximately the same length in the metric, we use the following techniques:

- mesh refinement and coarsening (h-method);
- change of nodal connectivity (diagonal swapping);
- relocalization of nodes at the 'centre' of their neighbours (r-method).

We purposely avoid using the *p-method*, which increases the degree of the basis functions in the finite element method (see Gui and Babuška [16] and Oden *et al.*[17]). It is hoped that the results presented in this series will make it abundantly clear that a low-order method, with a properly adapted or optimized mesh, can provide very accurate results.

3.1. Mesh refinement

The mesh is refined by sweeping over all the edges and by carrying out local modifications where needed. If the length $e(\gamma)$ of an edge in the given metric is greater than $L_{\max}$, then this edge is halved by introducing a new node in the middle (Figure 3).

Care must be taken, however, on boundary edges. The problem is that a mesh is not a perfect representation of the domain and the boundaries of the mesh are not the exact boundaries of the domain. When cutting a boundary edge by introducing a new node, this node must be reprojected onto the boundary of the domain, given by some computer aided design (CAD) description. In some cases, however, the projection of the new node onto the boundary may trap some interior nodes outside of the domain and these nodes must then be removed.

3.2. Mesh coarsening

The mesh is coarsened by sweeping over all the edges and by carrying out local modifications where needed. If the length $e(\gamma)$ in the given metric of an edge is lower than $L_{\min}$, then one of

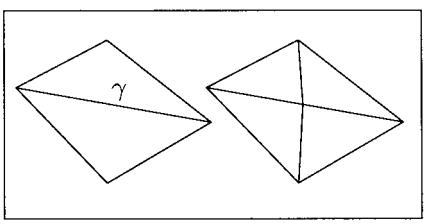

Figure 3. Edge refinement.

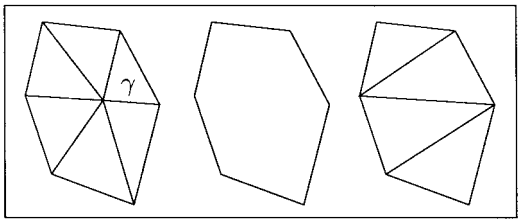

Figure 4. Edge coarsening.

the two nodes of the edge is removed, creating a 'hole' in the mesh, which is then filled by a triangulation process (Figure 4).

As in the previous section, special care must be taken with boundary edges.

3.3. Mesh reconnection

The nodes of the mesh are reconnected by sweeping over all the edges and carrying out local modifications where needed. An edge between two triangles is a diagonal of a quadrilateral (see Figure 5), and one has to choose between its two diagonals to get the 'best' triangles.

This *edge swapping* procedure is a classical way of building a Delaunay triangulation (see George [18]), which maximizes the minimum angles. To choose between the two configurations, we maximize the minimum of r/p, where r is the radius of the inscribed circle of the triangle and p is the half-perimeter, with $2p = a + b + c$, with a, b and c being the lengths of the edges of the triangle. By using the relation $S = rp$, where S is the area of the triangle, and by using the Heron formula $S^2 = p(p-a)(p-b)(p-c)$, a new criterion is defined

$$\text{shape}(\triangle) = 27\left(\frac{r}{p}\right)^2 = 27\,\frac{(p-a)(p-b)(p-c)}{p^3}. \tag{17}$$

This criterion is minimum and zero for a flat triangle and maximum and exactly one (with the scale factor 27) for an equilateral triangle. While this is no longer the Delaunay criterion, maximizing the minimum of the shape of the triangles according to criterion (17) will yield a triangulation very near the Delaunay one. Criterion (17) is used because it depends only on the

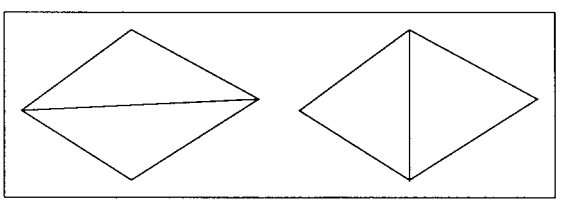

Figure 5. Node reconnection.

length of the edges of the triangles, which can be measured in the chosen metric. A mesh is therefore constructed where all triangles are approximately equilateral in the metric, thus having an error estimate equi-distributed over all the *edges* of the mesh. One should also note that in a non-Euclidean metric, this shape criterion could be negative.

One needs to exercise care in the implementation of this diagonal swapping process. Because criterion (17) depends on edge length, but not on orientation, the conformity must be checked before any diagonal swapping. If an interior node belongs to only three edges, none of these edges can be swapped. Moreover, if an edge belongs to two triangles that form a concave quadrilateral, this edge cannot be swapped.

3.4. Mesh movement

Moving nodes to smooth a mesh is a standard step in generation and adaptation codes. The most straightforward approach would sweep the nodes and relocalize them at the baricentre of their neighbours. In the current process, this approach would be anathema to the anisotropic nature of the mesh. Thus, it is necessary for the metric to be included in the mesh movement procedure.

The proposed strategy relies on an r-method, illustrated in Figure 6. The edges are viewed as a network of springs [19,20], whose stiffness constant is proportional to the edge error estimate. The ideal position of the vertices is then interpreted as the solution of an energy minimization problem. This yields for a node x_J, all others being fixed,

$$\min_{x_J \in \mathbb{R}^2} E(x_J) = \min_{x_J \in \mathbb{R}^2} \sum_{I=1}^{n} (x_I - x_J)^2 k_I(x_J), \tag{18}$$

where $k_I(x_J)$ is the spring constant between x_J and x_I. This constant should be independent of the edge length and for this purpose, we use

$$k_I(x_J) = e(x_I - x_J)/\|x_I - x_J\|, \tag{19}$$

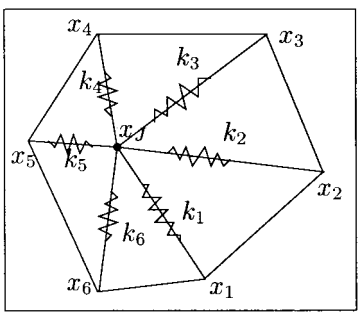

Figure 6. Spring analogy.

where $e(x_I - x_J)$ is the metric length of the edge $\overline{x_J x_I}$ (Equation (13)) and $\|x_I - x_J\|$ is its Euclidean length. This heuristic formula was chosen as it provided the best results among those tried. Taking the gradient of Equation (18), the problem to find a position x_J that minimizes the energy means finding a point x_J that is the root of

$$\sum_{I=1}^{n} F_I(x_J) = \sum_{I=1}^{n} (x_I - x_J) k_I(x_J) = 0, \tag{20}$$

where by Hooke's law, $(x_I - x_J)k_I(x_J)$ is the force $F_I(x_J)$ related to the edge $\overline{x_J x_I}$. It must be noted that this is a highly non-linear problem: when a node is moved, the edges change, along with the metric and spring constants. Equation (20) is solved node by node by a Gauss–Seidel method, with all other nodes remaining fixed. For one node, Equation (20) is solved by an iterative fixed point method, which is a weighted average

$$x_J^{\text{new}} = x_J^{\text{old}} + \omega \frac{\sum_{I=1}^{n} (x_I - x_J^{\text{old}}) k_I(x_J^{\text{old}})}{\sum_{I=1}^{n} k_I(x_J^{\text{old}})}, \tag{21}$$

where ω is a relaxation factor. It is futile to carry out many iterations of Equation (21) and find an accurate solution x_J because the neighbouring nodes x_I will also eventually move. Thus, it is more important to carry out many iterations of the overall Gauss–Seidel loop.

Some care has to be taken to prevent mesh degeneration or inverted elements, while moving nodes. For example, a node should not be allowed to cross neighbouring edges and each node should only be able to move only in the convex intersection of its neighbourhood. Mesh movement methods in the literature impose many controls on the procedure, with some using measures of smoothness or orthogonality, in addition to the error estimate, to control the quality of node distribution (see Nakahashi and Deiwert [21,22], Palmerio [5,23] and a review in Hawken *et al.* [24]). This can be done either by adding extra terms to the energy equation (18) or by limiting the node displacement to a smaller zone. Such constraints may be necessary in finite difference methods where a high degree of grid smoothness and orthogonality is required. For FEM, however, such constraints are unnecessary and nodes can move freely as long as elements remain convex.

For structured grids, moving nodes is the only available way of adapting the mesh, if connectivity is to remain fixed (see Ait-Ali-Yahia *et al.* [25]). The mesh movement scheme sketched above works well and is sufficient to give highly anisotropic meshes, as will be illustrated in Section 4.1.

For unstructured meshes, if one applies the spring analogy with mesh refinement, mesh coarsening and mesh reconnection, the results are not impressive and no convergence can usually be attained. The problem is that the three strategies attempt building a mesh with equilateral edges in the given metric, while the mesh movement through the spring analogy may not yield a mesh with equal edges. For example, suppose that all the support edges of a node have been made exactly of the same metric length and the spring analogy then applied. If the orientations of the edges are not uniform around the node (see Figure 7) then the node

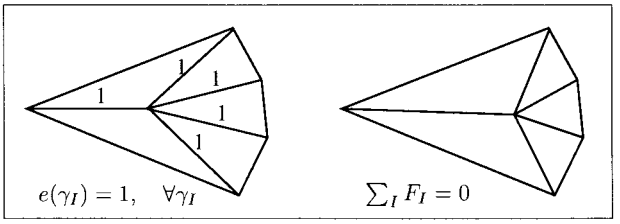

Figure 7. Spring analogy does not build a mesh with constant edge lengths.

will move to reach a position of equilibrium, but where the edges are not of the same metric length. In brief, the spring analogy does not work along the same spirit as mesh refinement, coarsening and reconnection.

The problem is overcome, again heuristically, by introducing a forcing term into Equation (20),

$$\sum_{I=1} F_I(x_J) \equiv \sum_{I=1}^{n} (x_I - x_J) k_I(x) = \sum_{I=1}^{n} (x_I - x_J) \overline{k_I}(x_J),$$

$$\text{with } \overline{k_I}(x_J) = \frac{1}{n} \frac{\sum_{I=1}^{n} e(x_I - x_J)}{\|x_I - x_J\|}, \tag{22}$$

to ensure that energy is measured locally by the discrepancy of the lengths from their averages. Equation (22) is solved by an iterative fixed point method, which is a weighted average

$$x_J^{\text{new}} = x_J^{\text{old}} + \omega \frac{\sum_{I=1}^{n} (x_I - x_J^{\text{old}})(k_I(x_J^{\text{old}}) - \overline{k_I}(x_J^{\text{old}}))}{\sum_{I=1}^{n} k_I(x_j^{\text{old}})}. \tag{23}$$

With this iterative method, if x_J is at the centre, i.e. all the edges around are of the same metric length, then $k_I(x_J) = \overline{k_I}(x_J)$ for all I. The numerator will vanish, but the denominator will not. Then x_J^{new} will be equal to x_j^{old} and the node will not move. This fixed point does not, however, necessarily correspond to a minimum of the energy. For unstructured meshes, unstable points are easily recovered by the other parts of the procedure.

3.5. Sequencing the operations

Particular attention has to be paid to the different criteria driving the adaptation process. For example, the refinement process converges in a few sweeps over all edges, and the same is true for mesh coarsening. But alternating refinement and coarsening can loop infinitely when the threshold value for cutting an edge into two, and that for removing an edge to create longer

ones, are too close to each other. So all the criteria governing the local operations must be sequenced in such a way that the overall process converges.

Node displacement is the most powerful tool in the current optimization strategy. It can be used solely when the mesh topology has to be kept unchanged, such as with structured grids or unstructured meshes with a fixed number of nodes. When combined with other techniques it greatly improves the mesh quality. In fact, all other processes are discontinuous, sporadic or binary: one chooses to do something or not depending on whether a criterion is above a threshold value. Displacement, however, is a continuous adaptive process that can perform surgical improvements after a discontinuous process.

After much experimentation, the following algorithm has been retained:

1. Smooth the mesh after estimating the error by alternatively:
   (a) swapping all the edges until convergence,
   (b) moving all the nodes iteratively.
2. Adapt the mesh by iterating over the following loop:
   (a) refine all edges above a threshold error estimate,
   (b) swap the edges until convergence, then apply node movement,
   (c) remove all nodes whose edges have an error estimate below a threshold value,
   (d) swap the edges until convergence, then apply node movement.
3. Finally, smooth the mesh by repeating loop (1) before solving the equations again, starting from an interpolated solution.

To conclude this section, there is an analogy between a CFD solver and a mesher. From a mesh (nodes and connectivity table), a CFD solver determines a solution (ρ, u, v, T, for example) which minimizes the residual of the equations to be solved. From a solution, the current mesher finds a mesh that minimizes the edge error estimate. Adaptation has been traditionally used to improve the mesh and the solution, but the process has not often been carried to convergence towards an optimal mesh. In fact, the current coupling, as will be demonstrated, yields not only adapted meshes but optimal ones in the chosen metric.

3.6. Coupling with a solver

The goal of the current approach is thus not only to obtain an improved mesh, but to converge the coupled solver–mesh adapter to an optimal solution, on an optimized mesh. This is performed by coupling them in the following way:

> Given (M_n, S_n), a mesh and a solution on this mesh at step n, the mesher produces a new mesh M_{n+1} and a solution $S_{n+1/2}$, the reinterpolation of S_n on M_{n+1}. A solution S_{n+1} on M_{n+1} is then obtained with the solver starting with $S_{n+1/2}$ as an initial guess. The iterations go over until convergence is reached.

The following points are important when this procedure is applied:

- Instead of fully converging the solution on intermediate meshes, it is better to do more overall loops with partial resolution on them. A close coupling provides a maximum of flexibility to the mesh and permits one to follow the evolution of the solution. This is done through frequent adaptation. Typically, about a few dozen adaptive steps may have to be

used to reach a converged steady flow, starting from a uniform initial solution. Good results can also be obtained for unsteady flows (with steady shocks), provided a stronger level of convergence is achieved during the solution steps.

- Many stretched elements must be concentrated in the area of shock waves, to capture them efficiently. As a result, to pass from an adapted solution at a given *Re* to an adapted one at another *Re* may not be a good practice. Adapting the mesh is like 'freezing' the shock wave. It is better to restart the computation on a generic mesh, and use a high artificial viscosity during the first loops, reducing it as the mesh converges. In fact, flows over an NACA 0012 airfoil with Reynolds numbers as large as 32000 were adaptively computed this way (see Section 4 below and Bourgault [26] for more details), with absolutely no artificial viscosity in the final steps, demonstrating that, with unequal order or mixed-type finite elements, a good mesh can lead to solutions requiring no artificial viscosity.
- The choice of the flow variable used to control the mesher is not unique. With the variable chosen, one tries to directionally equalize the error for all the areas of variation, for all the variables. The local Mach number appears to be a good choice to track shock waves, as well as boundary layers. A more appropriate choice could be a *scaled* sum of all the variables [27]. Castro-Díaz *et al.* [13] proposed an extension of the metric definition to systems by taking the intersection of all the ellipses associated with the metric based on each and every variable of the system. This 'intersection metric' as well as the metric based on the scaled sum of variables stated above may be good ideas, although their ability to always produce highly anisotropic grids is not guaranteed: averaging directional errors usually ends up with isotropic information.

In the current masher–solver loop, only a mesh and solution files are exchanged between the mesher and the solver. Hence, the adaptive process can be used generically to improve the solution quality of a wide variety of computational problems, using any discretization technique that allows the evaluation of an edge error estimate.

4. NUMERICAL RESULTS

The capability of the proposed MOM is illustrated through numerical examples. An extensive validation of MOM against analytical, numerical and experimental data will follow in Parts II and III of the paper for structured and unstructured meshes.

4.1. Hypersonic inviscid flows, on structured grids

The mesh adaptation method is tested on a structured quadrilateral grid for a hypersonic flow field. The flow is modelled by the two-dimensional Euler equations written in conservative form and solved by an implicit, Galerkin finite element method. The primitive variables $[\rho, u, v, e]$ are interpolated by bilinear shape functions. Artificial dissipation terms, necessary for preventing instabilities, are added in the form of Laplacian conservative variables $[\rho, \rho u, \rho v, \rho h]$ to the governing equations. More details about the flow solver may be found in Ait-Ali-Yahia *et al.* [28].

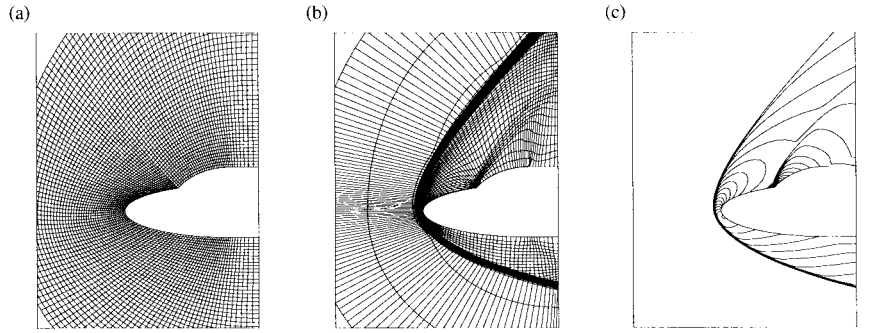

Figure 8. Mach number = 15 and 30° angle of attack flow over a double ellipse: (a) initial grid, (b) final adapted grid and (c) final adapted solution in temperature contours.

Since the present flow solver has no capability of handling elements with hanging nodes, only the mesh movement scheme (see Section 3.4) is used during the adaptation process.

The proposed methodology is applied to a Mach 15 inviscid flow over a double ellipse at a 30° angle of attack [29]. The flow field contains a *strong* detached shock wave, followed by a *weak* canopy shock, making this test quite challenging for flow solvers.

The calculations were initiated on a 45×124 grid, which is depicted in Figure 8(a). The final adapted grid, shown in Figure 8(b), required six cycles of adaptation and the corresponding adapted solution, displayed in Figure 8(c) as temperature contours, demonstrates the benefits of mesh adaptation in efficiently resolving multiple shocks of different strengths.

In the current test case, both flow solver and grid adaptation procedures are placed in an iterative loop, which is repeated until the lowest value of a user-specified artificial dissipation coefficient is reached. For each adaptation cycle, the L_2 norm of the solver's residual is lowered by two orders of magnitude and then the mesh nodes are displaced a few hundred times. It is worth mentioning here that these excellent results are obtained despite the fact that the FEM code was only used in *first-order accuracy* mode.

4.2. Transonic and supersonic viscous flows, on unstructured grids

The Navier–Stokes finite element solver used in this section is based on the non-conservative form of the equations for the primitive variables. The thermodynamic variables, i.e. density and temperature, are discretized with linear elements and the velocity field with quadratic, or P1-*iso*-P2, elements. *Surprisingly, no explicitly introduced numerical viscosity is needed to stabilize low-to-moderate Reynolds number solutions on the final adapted meshes.* As a result, all the computations over the NACA 0012 shown here have been obtained without any artificial viscosity. However, for other convection-dominated flows at high Reynolds number, an anisotropic viscous term with an oscillation detector had to be added to damp instabilities around shocks. For these situations, we tried to keep the artificial viscosity as low as possible, decreasing it gradually as the resolution–remeshing loop progressed. Time marching is done implicitly with a Gear scheme for both steady and unsteady flows. The system of equations is

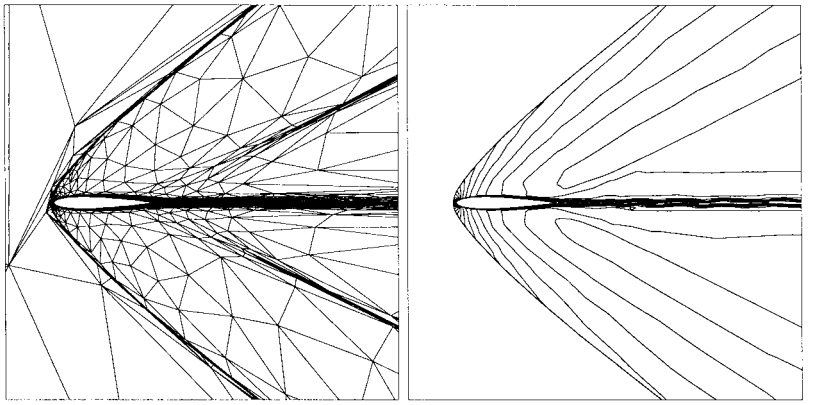

Figure 9. Adapted mesh with 4798 nodes for $M_\infty = 2$ and $Re_\infty = 32000$ over an NACA 0012. Local Mach numbers are plotted with $\Delta M = 0.1$.

solved by the non-linear GMRES method [30], with diagonal preconditioning. For more details on the solver, see Bourgault [26] or Boivin and Fortin [31,32].

Two typical flows over an NACA 0012 airfoil were used as test cases. The first is a supersonic flow at $M_\infty = 2$ and $Re_\infty = 32000$. Figure 9 shows a steady flow almost everywhere, except in the wake, where a small periodic instability appears. Two shock waves are observed, a detached bow shock in front of the airfoil and one induced by the wake at the trailing edge. The latter, being weaker, needs more adaptation steps than the bow shock in order to be properly resolved.

The second test case is that of a transonic flow at $M_\infty = 0.85$ and $Re_\infty = 10000$. This flow is clearly unsteady showing a large von Karman alley. Our adaptation strategy, by being an *a posteriori* one, cannot predict in advance the movement of the vortices. Thus, instead of doing frequent remeshing, which can introduce extra numerical diffusion due to the reinterpolation of solutions between meshes, we prefer to use triangles smaller than the smallest vortices in the wake. To do so, a global optimal mesh size h, which controls the remesher, is set to a smaller value than in the previous example. As can be seen in Figure 10, shock waves right above and below the airfoil result from the supersonic depression zone. Figure 11 shows a magnification of the small isotropic triangles obtained with the remesher in the area where the shock wave meets the boundary layer.

5. CONCLUSIONS

The general principles of a novel and cost-effective anisotropic mesh adaptation methodology have been laid out in the present paper, considered Part I of a three-part series. The error estimator has been discussed and its efficiency demonstrated on structured and unstructured flow problems with shocks and wakes. The anisotropy of the resulting grids is clearly evident,

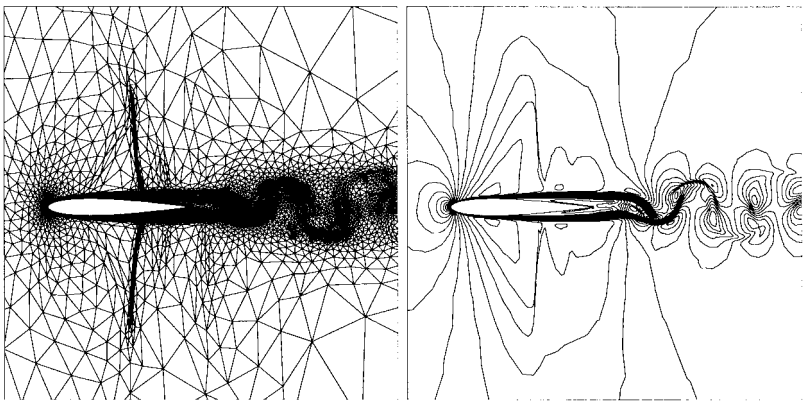

Figure 10. Adapted mesh with 9959 nodes for $M_\infty = 0.85$ and $Re_\infty = 10000$ over an NACA 0012. Local Mach numbers are plotted with $\Delta M = 0.05$.

as well as the quasi-impossibility for the user to have 'divined', at grid generation time, that these were the most appropriate grids. Thus, it is conclusively shown that it is bad practice to try to generate grids based on intuition and that anisotropic grids generated by a full-coupling of solver and adaptation is the most logical way to obtain appropriate grids and to increase the accuracy and reliability of CFD calculations.

In Part 2, addressing structured and unstructured grids, a thorough validation of the adaption algorithm is undertaken. In particular it will be shown that the algorithm is

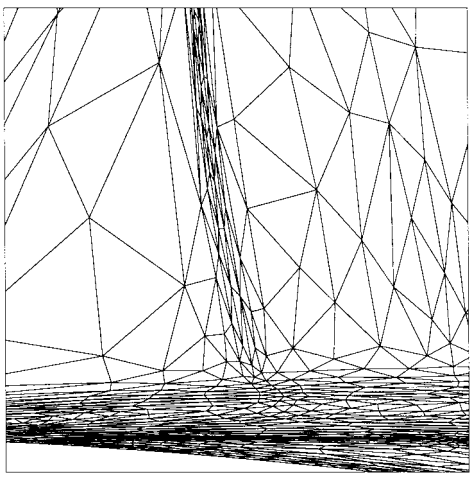

Figure 11. Zoom of the mesh of Figure 10 in the shock–boundary layer interaction zone.

reversible. This means that, starting from highly distorted triangular and rectangular grids defined on triangular or rectangular domains, the algorithm is able to recover a perfectly uniform mesh if a uniform second derivative is specified. Results will also be presented for the well-known AGARD series of test cases, ranging from 0.8 to 1.2, and compared with solutions obtained with other methods. The cost-effectiveness of the present approach will be evident.

Finally, in Part 3, the full power of the approach will be unleashed for three-dimensional problems. While the greater flexibility of these grids is acknowledged, this is usually tempered by the accompanying loss of accuracy. It will be shown that not only does mesh adaptation palliate this deficiency but surprisingly, the final adapted grid and solution turn out to be the same, independent of the starting grid. An equally important surprise is that the grid and the solution, at least in the tested runs, turn out to be independent of the solution algorithm (or even commercial code) used. This at least demonstrates heuristically that many of the advantages shown by higher accuracy schemes show up mostly when grids are inappropriate, but that adapted grids are impervious to showcasing such advantages. CAD fidelity during adaptation will also be demonstrated.

ACKNOWLEDGMENTS

The authors would like to thank NSERC and FCAR for Operating and Strategic grants under which this work was supported and the CRM (Centre de Recherches Mathématiques de l'Universite de Montréal) for its partial support of a Post-Doctoral Fellowship to Dr J. Dompierre. Thanks are also due to Professor Sylvain Boivin of the Université du Québec à Chicoutimi, for allowing the use of his P1/(P1-iso-P2) Navier–Stokes solver [31,32].

REFERENCES

1. J.T. Oden, T. Strouboulis and P. Devloo, 'Adaptive finite element methods for high-speed compressible flows', *Int. J. Numer. Methods Fluids*, **7**, 1211–1228 (1987).
2. Y. Kallinderis, 'Adaptive hybrid prismatic/tetrahedral grids', *Int. J. Numer. Methods Fluids*, **20**, 1023–1037 (1995).
3. R.L. Davis and J.F. Dannenhoffer, 'Decomposition and parallelization strategies of adaptive grid-embedding techniques', *Int. J. Comput. Fluid Dyn.*, **1**, 79–93 (1993).
4. J. Peraire, M. Vahdati, K. Morgan and O.C. Zienkiewicz, 'Adaptive remeshing for compressible flow computations', *J. Comput. Phys.*, **72**, 449–466 (1987).
5. B. Palmerio, 'A two-dimensional FEM adaptive moving-node method for steady Euler flow simulation', *Comput. Methods Appl. Mech. Eng.*, **71**, 315–340 (1988).
6. R. Körnhuber and R. Roitzsch, 'On adaptive grid refinement in the presence of internal or boundary layers', *Impact Comput. Sci. Eng.*, **2**, 40–72 (1990).
7. M.-G. Vallet, 'Génération de maillages éléments finis anisotropes et adaptatifs', *Ph.D. Thesis*, Université Pierre et Marie Curie, Paris VI, France, 1992.
8. M. Fortin, M.-G. Vallet, D. Poirier and W.G. Habashi, 'Error estimation and directionally adaptive meshing', *25th AIAA Fluid Dynamics Conference*, No. AIAA-94-2211, Colorado Springs, CO, June 1994.
9. M. Fortin, Y. Bourgault, W.G. Habashi, J. Dompierre and M.-G. Vallet, 'Mesh adaptation for viscous compressible flows', in M.M. Cecchi, K. Morgan, J. Periaux, B.A. Schrefler and O.C. Zienkiewicz (eds.), *Ninth International Conference on Finite Elements in Fluids, New Trends and Applications*, Venezia, Italy, IACM, October 1995, pp. 1151–1160.
10. W.G. Habashi, M. Fortin, J. Dompierre, M.-G. Vallet, D. Ait-Ali-Yahia, Y. Bourgault, M.P. Robichaud, A. Tam and S. Boivin, 'Anisotropic mesh optimization for structured and unstructured meshes', *28th Computational Fluid Dynamics Lecture Series*, von Karman Institute for Fluid Dynamics, March 1997.
11. M.J. Castro-Díaz and F. Hecht, 'Error interpolation minimization and anisotropic mesh generation', in M. Morandi Cecchi, K. Morgan, J. Périaux, B.A. Schrefler and O.C. Zienkiewicz (eds.), *Ninth International Conference on Finite Elements in Fluids, New Trends and Applications*, Venezia, Italy, IACM, October 1995, pp. 1139–1148.

12. M.J. Castro-Díaz, F. Hecht and B. Mohammadi, 'Anisotropic grid adaptation for inviscid and viscous flows simulations', *4th International Meshing Roundtable*, Albuquerque, NM, October 1995, pp. 73–85.
13. M.J. Castro-Díaz, F. Hecht, B. Mohammadi and O. Pironneau, 'Anisotropic unstructured mesh adaptation for flow simulations', *Int. J. Numer. Methods Fluids*, **25**, 475–491 (1997).
14. E.F. D'Azevedo and R.B. Simpson, 'On optimal interpolation triangle incidences', *SIAM J. Sci. Stat. Comput.*, **10**, 1063–1075 (1989).
15. E.F. D'Azevedo and R.B. Simpson, 'On optimal triangular meshes for minimizing the gradient error', *Numer. Math.*, **59**, 321–348 (1991).
16. W. Gui and I. Babuška, 'The h, p and h–p version of the finite element method in one dimension. Part III: the adaptive h–p version', *Numer. Math.*, **48** (1986).
17. J.T. Oden, T. Liszka and W. Wu, 'A h–p adaptive finite element method for incompressible viscous flows', in *The Mathematics of Finite Elements and Applications VII*, Academic Press, New York, 1991.
18. P.-L. George, *Automatic Mesh Generation: Application to Finite Element Methods*, Wiley, New York, 1991.
19. P.A. Gnoffo, 'A finite volume, adaptive grid algorithm applied to planetary entry flow fields', *AIME J.*, **21**, 1249–1254 (1983).
20. P.A. Gnoffo, 'A vectorized finite volume, adaptive grid algorithm for Navier–Stokes calculation', in J.F. Thompson (ed.), *Numerical Grid Generation*, No. 9, Elsevier–North Holland, New York, 1982, pp. 819–835.
21. K. Nakahashi and G.S. Deiwert, 'Three-dimensional adaptive grid method', *AIAA J.*, **24**, 948–954 (1985).
22. K. Nakahashi and G.S. Deiwert, 'Self-adaptive grid method with application to airfoil flow', *AIAA J.*, **25**, 513–520 (1987).
23. B. Palmerio, 'An attraction-repulsion mesh adaption model for flow solution on unstructured grids', *Comput. Fluids*, **23**, 487–506 (1994).
24. D.F. Hawken, J.J. Gottlieb and J.S. Hansen, 'Review of some adaptive node-movement techniques in finite element and finite difference solutions of partial differential equations', *J. Comput. Phys.*, **95**, 254–302 (1991).
25. D. Ait-Ali-Yahia, W.G. Habashi, A. Tam, M.-G. Vallet and M. Fortin, 'A directionally adaptive methodology using an edge-based error estimate on quadrilateral grids', *Int. J. Numer. Methods Fluids*, **23**, 673–390 (1996).
26. Y. Bourgault, 'Méthodes des éléments finis en mécanique des fluides: conservation et autres propriétés', *Ph.D. Thesis*, Université Laval, Québec, Canada, 1996.
27. R. Löhner, 'Adaptive remeshing for transient problems', *Comput. Methods Appl. Mech. Eng.*, **75**, 195–214 (1989).
28. D. Ait-Ali-Yahia, 'A finite element segregated method for hypersonic thermo-chemical equilibrium and nonequilibrium flows using directionally-adaptive meshes', *Ph.D. Thesis*, Department of Mechanical Engineering, Concordia University, Montreal, Quebec, Canada, 1996.
29. R. Abgrall, J.-A. Désidéri, R. Glowinski, M. Mallet and J. Periaux (eds.), *Hypersonic Flows for Reentry Problems*, vol. 3, Springer, Berlin, 1991.
30. P. Brown and Y. Saad, 'Hybrid Krylov methods for nonlinear systems of equations', *Tech. Rep. UCRL-97645*, Lawrence Livermore National Laboratory, November 1987.
31. S. Boivin and M. Fortin, 'A new artificial viscosity method for compressible viscous flow simulations by FEM', *Int. J. Comput. Fluid Dyn.*, **1**, 25–41 (1993).
32. S. Boivin and M. Fortin, 'A nonisotropic artificial viscosity method: application to the simulation of compressible viscous flows', *Int. J. Comput. Fluid Dyn.*, **7**, 327–338 (1996).

